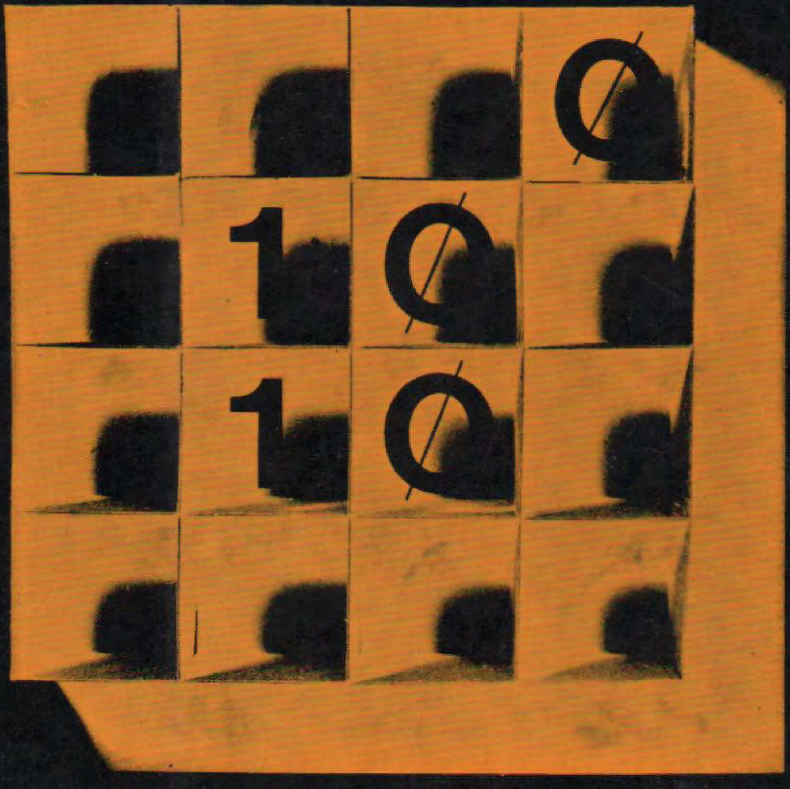
A. PETITCLERC

TRATADO DE ORDENADORES

TOMO II

principales elementos y organización de un ordenador



editores técnicos asociados, s. a.
BARCELONA

Tratado de ordenadores

TOMO 2

Principales elementos y organización de un ordenador

A. PETITCLERC
Ingeniero de la Compañía IBM Francia
Miembro de la "Société Française des Électroniciens et des Radioélectriciens"

Tratado de ordenadores

TOMO 2

Principales elementos y organización de un ordenador

editores técnicos asociados, s. a.
Maignón, 26 – Barcelona - 12 – España
1 9 7 4

Primera edición española, traducida del francés por
EDUARDO BUZZANCA CAÑELLAS
Dr. Ingeniero Industrial

Título de la obra original:

TRAITE DES ORDINATEURS

Tome 2

Principaux eléments et organisation d'un ordinateur

Publicada por

DUNOD — París

© EDITORES TECNICOS ASOCIADOS, S.A. — Barcelona 1974

Depósito legal B. 22.236 - 1974
ISBN (obra completa) 84-7146-141-2
ISBN (tomo 2) 84-7146-143-9

Impreso en España
Printed in Spain

ROMARGRAF, S.A. — Sants, 387 — Barcelona - **España**

Indice de materias

Prólogo IX

Capítulo 1. ***Principales elementos del ordenador*** .. 1

Introducción 3
Los codificadores, decodificadores y transcodificadores 3
Codificador decimal ⇒ decimal codificado binario 3
Decodificador BCD — decimal 5
Caso de códigos incompletos 15
Los transcodificadores 16
Los contadores 20
Determinación analítica de los contadores 20
Contador binario puro. Utilización de las básculas R - S 20
Contadores que utilizan las básculas R - S - T 28
Contadores que utilizan las básculas T. Contador binario puro 33
Contadores que utilizan las básculas J - K 35
Contador binario puro que utiliza las básculas J - K 36
Contadores especiales 41
Contadores decimales 50
Contadores en anillo y registros de desplazamientos 53
Contadores de funcionamiento dinámico 61
Registros de desplazamientos dinámicos 63
Contador decimal biquinario 65
Transferencia de las informaciones en un calculador 66
Simbolismo utilizado 66
Transferencias paralelas. Utilización de las básculas R - S 67
Transferencia paralela por báscula T 71
Adición lógica entre dos registros paralelos 72
Multiplicación lógica entre dos registros paralelos 73
"O exclusivo" entre dos registros 75
Transferencia paralela con conversión según el valor del signo 76
Transferencia en serie de una información de un registro a otro 77
Los circuitos de complementación 82

Capítulo 2. ***Operadores aritméticos*** 87

Diferentes tipos de adicionadores y sustractores 89
El semiadicionador 89
El semisustractor 90
Adicionador completo 91

Sustractor completo 93
Adicionador-sustractor 94
Adicionador-sustractor que realiza las operaciones lógicas "Y" y "O exclusivo" 95
Adicionador completo realizado únicamente con circuitos NOR y NAND 96
Adicionador-sustractor de básculas 98
El acumulador 102
Adicionador-sustractor en serie 102
Adicionadores decimales 107
Los comparadores 116
Los transcodificadores en serie 122

Capítulo 3. *La unidad de memoria* 127

Introducción 129
Memoria de núcleos magnéticos 129
Selección de un núcleo 131
Memoria tridimensional 133
Selección de una palabra en la memoria 134
Realización del registro memoria 136
Realización del registro y del decodificador de direcciones 138
Realización práctica del sistema de direccionamiento 139
Memorias de tres hilos 140
Memorias de selección lineal 140
Memorias de dos hilos 142
Memorias muertas 142
Memorias de diodos túnel 144

Capítulo 4. *La unidad aritmética y lógica* 149

Introducción 151
Constitución de la unidad aritmética 151
Máquinas en paralelo y máquinas en serie 152
Unidad aritmética y lógica en paralelo 153
Adición y sustracción 153
Caso de números representados en valor absoluto y con signo 153
Caso de números negativos representados por sus complementos a 2 160
La multiplicación 163
Números representados en valor absoluto y con signo 164
El multiplicador de Booth 167
Segunda versión 171
Multiplicador rápido de productos simultáneos 176
La división 178
División por restauración de los restos parciales 178
División por el método de comparación del signo del resto parcial y del divisor 183

Operaciones lógicas "Y" y "O exclusivo" 194
Las máquinas en serie 196
Adición y sustracción 196
Caso de números representados en valor absoluto y con signo 196
Caso de números negativos representados por sus complementos a 2 198
Multiplicación en serie 199
Números representados en valor absoluto y con signo 199
Números negativos representados por sus complementos a 2 200
Divisor en serie 201
Números representados en valor absoluto y con signo 201
Unidad aritmética en coma flotante 203
Las máquinas decimales 206

Capítulo 5. ***Organización de un ordenador*** 221

Introducción 223
Formato de una instrucción 223
Diferentes categorías de instrucciones 225
Modificación de instrucciones 229
Instrucciones de entradas y salidas 231
Máquinas de varias direcciones 232
Concepción de un calculador 233
Circuitos de control de las micro-operaciones 236
Realización de las operaciones de bifurcación 245
Realización de las operaciones de desplazamiento 246
Estudio de un pequeño ordenador 251
Realización de los diferentes elementos 261
Extensión del código de operación 262
Tendencias en la concepción de los ordenadores modernos 262

Anexo 263

Bibliografía 265

Símbolos y abreviaturas

Una letra mayúscula designa en general un elemento del ordenador: báscula, registro o contador. Así *A* designa una báscula o un registro *A, C* designa un contador. Para indicar el contenido de estos elementos, éstas se pondrán entre paréntesis: *(A)* representa el contenido de la báscula o del registro; *A, (C)* el contenido del contador *C.*

Una flecha ⇒ indica una transferencia de izquierda a derecha. Así *(A)* ⇒ *B* indica la transferencia del contenido de *A* a *B, (C) − 1* ⇒ *C* indica que el contador *C* ha disminuido en una unidad, *(A) + (B)* ⇒ *B* indica que se efectúa la suma lógica del contenido de las dos básculas o de los dos registros *A* y *B* transfiriendo el resultado a *B.*

Ciertas básculas o registros que tienen funciones particulares se designan por una letra mayúscula seguida de una abreviatura: B Rep. designa la báscula report, B, OP. la báscula operación, R Op. el registro operación, C I el contador de instrucciones, R Dir. el registro de direcciones, C Dpl. un contador de desplazamientos, B Overf. designa la báscula de rebasamiento de capacidad. Hemos utilizado en las abreviaturas el término anglosajón overflow para designar el rebasamiento de capacidad, así como la abreviatura *BCD* para el código *DCB* (decimal codificado binario).

Introducción

Esta obra se propone describir la teoría y los principios generales de funcionamiento de los ordenadores. Se dirige a los ingenieros y técnicos de diversas disciplinas que deseen adquirir un profundo conocimiento sobre la materia, tanto en el dominio de los circuitos del ordenador como en su concepción. El lector de esta obra no necesita un conocimiento especial sobre informática. Se ha redactado con un deseo; ser lo suficientemente simple para poder utilizarse sin profesor. Su carácter didáctico permitirá que los lectores no técnicos que deseen más claridad en este dominio, lo aborden sin dificultades.

El principiante que inicia por primera vez el estudio de un ordenador sólo dispone de obras de divulgación o muy especializadas, que tratan de forma superficial o demasiado teórica un aspecto muy particular del funcionamiento de un ordenador y de su utilización. Resulta muy difícil realizar la síntesis entre los diferentes elementos del ordenador. Nos proponemos cubrir esta laguna dando a los lectores un texto que trate de forma detallada todas las principales funciones de un ordenador para iniciar a continuación el estudio de su organización interna y su utilización mediante un programa. Los conocimientos de electrónica se reducen al mínimo, el estudio de los circuitos se efectúa de forma lógica sin entrar en el detalle técnico de su realización. Esto es objeto de un curso de semiconductores. El lector que esté interesado por este dominio podrá consultar los excelentes tratados que sobre este tema existen en el mercado.

El ordenador tiene un lugar cada vez más importante en nuestra sociedad. Guía los aviones y satélites, controla los viajes interplanetarios, el tráfico de nuestras carreteras, la producción de nuestras fábricas, asegura la gestión en nuestras empresas. El día de mañana enseñará a nuestros hijos las materias más difíciles. Esta invasión del ordenador en nuestra vida cotidiana modificará las actividades de muchos de nosotros. Algunos construirán estos ordenadores, otros trabajarán estableciendo los programas específicos a cada problema que el ordenador deba resolver.

La presente obra, dividida en dos tomos, es el contenido de un curso que el autor da a los ingenieros y técnicos de la compañía I.B.M. en Francia para el perfeccionamiento científico de su personal.

En el tomo 1 se hallará la exposición de los métodos de análisis de los circuitos de conmutación y, en el tomo 2, el estudio de los circuitos de un ordenador y de su organización interna. El primer capítulo del tomo 1 se dedica al estudio del álgebra de Boole, la herramienta por excelencia de los informáticos. Numerosos ejemplos ilustran el método. El segundo capítulo se dedica al estudio de los diagramas de Karnaugh, que representan un medio poderoso y elegante de simplificación de los circuitos lógicos más complejos. Se estudian otros métodos tales como los de Quine y Mc. Cluskey. El tercer capítulo trata la aritmética binaria. El lector hallará en este capítulo el desarrollo de los algoritmos de las distintas operaciones realizadas por el ordenador.

El tomo 2 trata las principales funciones del ordenador y sus circuitos. El capítulo primero se dedica al estudio de los codificadores, contadores y registros, así como los circuitos que transfieren la información de registro a registro en serie o en paralelo. El capítulo segundo estudia con mucho detalle los diferentes tipos de adicionadores y substractores así como las importantes funciones del acumulador. El tercer capítulo estudia la memoria del ordenador. Se dedica una atención especial a la determinación de los circuitos de selección y direccionamiento que generalmente es un punto difícil para muchos de los principiantes. El siguiente capítulo se dedica a la unidad aritmética y lógica, particularmente a los circuitos que realizan los ciclos de instrucción y ejecución de las principales operaciones aritméticas. El tomo 2 termina con un capítulo dedicado a la organización interna del ordenador. El lector puede seguir de esta manera, a partir de un código previamente definido, la codificación de las instrucciones y su ejecución en los diferentes registros y circuitos del ordenador. Terminado el tomo 2, el lector estará en condiciones de conocer el funcionamiento de un ordenador y la forma de determinar sus circuitos.

El autor da las gracias a la compañía I.B.M. Francia, por la favorable acogida dispensada a la obra, así como a todos sus alumnos por el interés testimoniado continuamente durante la enseñanza de este curso.

A. PETITCLERC

Capítulo 1

Principales elementos del ordenador

1. Introducción

Examinaremos en este capítulo los principales órganos funcionales de un ordenador. En los capítulos siguientes estos órganos funcionales serán utilizados para el estudio y realización de la unidad aritmética, así como en la organización completa del ordenador.

Los órganos funcionales que estudiaremos son los siguientes:

– los codificadores, decodificadores y transcodificadores,
– los diferentes tipos de contadores,
– los circuitos de transferencia entre registros,
– los circuitos de complementación.

Estos diferentes órganos funcionales se encuentran en la máquina en gran cantidad. Resulta, pues, necesario conocer bien el funcionamiento de estos elementos antes de abordar la estructura interna del ordenador.

2. Los codificadores, decodificadores y transcodificadores

Se llama codificador y decodificador a un dispositivo que permite expresar una cantidad decimal en su equivalente binario y viceversa. Los codificadores están generalmente realizados con circuitos "O", los decodificadores con circuitos "Y". Los transcodificadores son circuitos que permiten expresar una cantidad codificada en un código diferente. Se trata, pues, de un sistema que permite cambiar de código.

Examinemos primeramente el caso de los codificadores por medio de un ejemplo: el codificador decimal ⇒ decimal codificado binario.

2.1 Codificador decimal ⇒ decimal codificado binario

Se desea expresar con este dispositivo, los 10 digitos (0 a 9) del sistema decimal en sus equivalentes binarios en el código BCD 1.2.4.8. El cuadro de certeza del codificador está representado en la figura 1. Se dispone de cuatro pares de salidas, correspondiendo cada una a los dos valores posibles 0 y 1 de un peso del código BCD. El esquema de este codificador está representado en la figura 2.

Cuando la línea que corresponde al valor decimal 5 es sensibilizada, aparece en la salida "verdad" del codificador 0101 y sobre la salida complementaria 1010. Si se sensibiliza la línea 9, tendremos en la salida "verdad" el valor binario 1001 y en la salida complementaria 0110. Si se atribuyen los símbolos W, X, Y, y Z a los pesos binarios 2^3, 2^2, 2^1 y 2^0 tendremos las relaciones siguientes:

$$W = 8 + 9, X = 4 + 5 + 6 + 7, Y = 2 + 3 + 6 + 7, Z = 1 + 3 + 5 + 7 + 9.$$

La posición 0 no se codifica.

Valor decimal	2^3		2^2		2^1		2^0	
	0	1	0	1	0	1	0	1
0	1	0	1	0	1	0	1	0
1	1	0	1	0	1	0	0	1
2	1	0	1	0	0	1	1	0
3	1	0	1	0	0	1	0	1
4	1	0	0	1	1	0	1	0
5	1	0	0	1	1	0	0	1
6	1	0	0	1	0	1	1	0
7	1	0	0	1	0	1	0	1
8	0	1	1	0	1	0	1	0
9	0	1	1	0	1	0	0	1

Fig. 1 *Cuadro de certeza del codificador decimal ⇒ BCD*

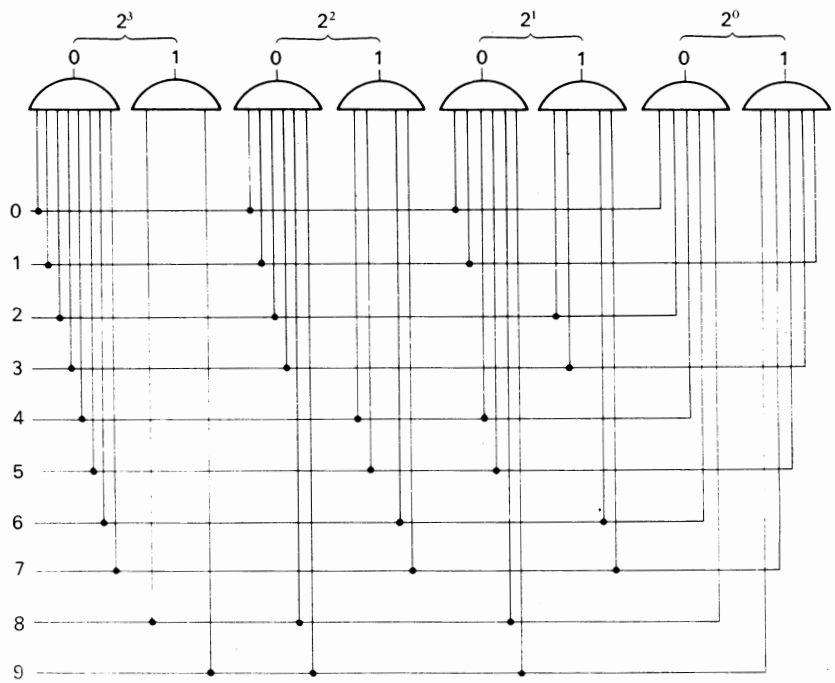


Fig. 2. *Codificador decimal ⇒ binario (BCD)*

2.2 Decodificador BCD ⇒ decimal

En este tipo de circuito la entrada está formada de 2 x 4 hilos correspondientes a los cuatro pesos del código BCD. Con este dispositivo se pueden decodificar $2^4 = 16$ valores decimales de 0 a 15. En realidad solamente se utilizan 10, de 0 a 9. Las combinaciones correspondientes a los valores decimales 10 a 15 *no son utilizados.*

Estas representan las condiciones prohibidas que se emplean para simplificar el circuito. El cuadro de certeza está representado en la figura 3.

Se han representado (Fig. 4) todas las salidas en un diagrama de Karnaugh en el que se han introducido las condiciones imposibles $\emptyset$. Se notará que las salidas P_2 a P_9 pueden simplificarse incorporando estas condiciones prohibidas en los bucles de simplificación. El índice de los términos P corresponde a la decodificación del valor decimal correspondiente.

De este modo, tendremos un 1 en P_9 cuando haya sobre las entradas "verdad" el número binario 1001. Debe notarse igualmente que cada una de las salidas del decodificador corresponde a un término P de las diferentes combinaciones de las cuatro variables *A, B, C* y *D.*

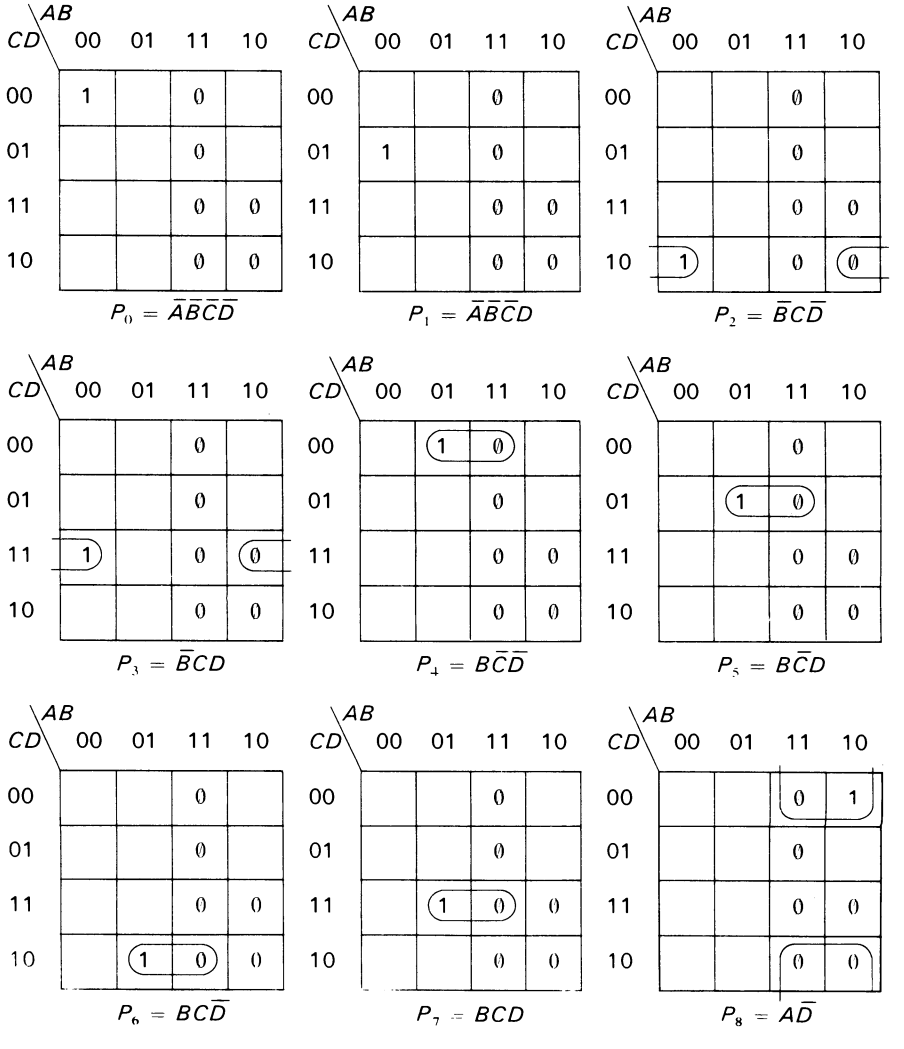
A	*B*	*C*	*D*	Salidas
0	0	0	0	$P_0 = \overline{A}\overline{B}\overline{C}\overline{D}$
0	0	0	1	$P_1 = \overline{A}\overline{B}\overline{C}D$
0	0	1	0	$P_2 = \overline{A}\overline{B}C\overline{D}$
0	0	1	1	$P_3 = \overline{A}\overline{B}CD$
0	1	0	0	$P_4 = \overline{A}B\overline{C}\overline{D}$
0	1	0	1	$P_5 = \overline{A}B\overline{C}D$
0	1	1	0	$P_6 = \overline{A}BC\overline{D}$
0	1	1	1	$P_7 = \overline{A}BCD$
1	0	0	0	$P_8 = A\overline{B}\overline{C}\overline{D}$
1	0	0	1	$P_9 = A\overline{B}\overline{C}D$
1	0	1	0	
1	0	1	1	Condiciones
1	1	0	0	prohibidas
1	1	0	1	
1	1	1	0	Salidas no utilizadas
1	1	1	1	

Fig. 3. *Cuadro de certeza del decodificador BCD ⇒ decimal*

La lectura de los diagramas de Karnaugh da:

$$P_0 = \bar{A}\bar{B}\bar{C}\bar{D} \qquad P_5 = B\bar{C}D$$
$$P_1 = \bar{A}\bar{B}\bar{C}D \qquad P_6 = BC\bar{D}$$
$$P_2 = \bar{B}C\bar{D} \qquad P_7 = BCD$$
$$P_3 = \bar{B}CD \qquad P_8 = A\bar{D}$$
$$P_4 = B\bar{C}\bar{D} \qquad P_9 \stackrel{\bullet}{=} AD\,.$$

El esquema del decodificador está representado en la figura 5.



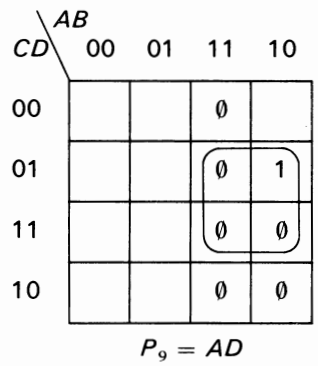


Fig. 4. *Diagramas de Karnaugh para cada una de las salidas del decodificador BCD → decimal*

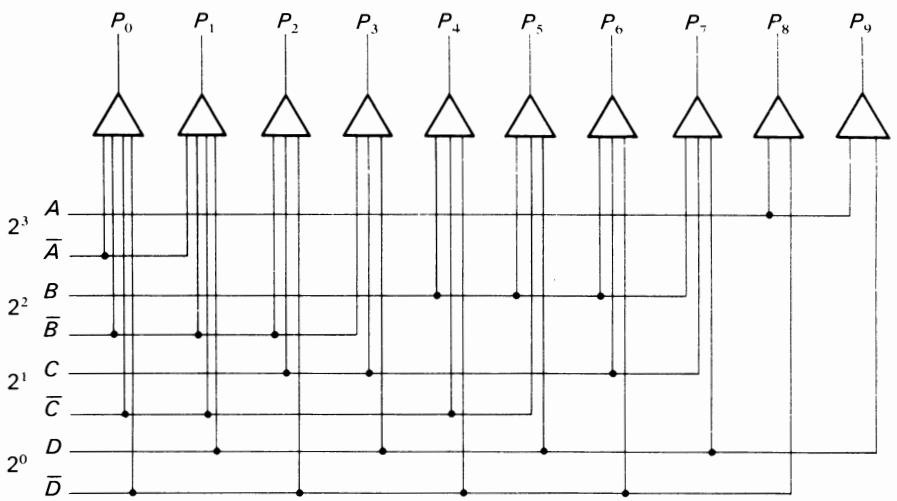


Fig. 5 *Esquema del decodificador binario ⇒ decimal*

2.2.1. *Caso de un decodificador en el que se utilizan todas las combinaciones.*

En el código más general en el que todas las combinaciones de entrada son utilizadas, habrán *n* pares de hilos de entrada y 2^n salidas. Así, en un codificador de seis entradas (2 x 3) tendremos a la salida $2^3 = 8$ valores decimales (0 a 7). En este caso se realiza para la decodificación, lo que se llama una "*matriz rectangular*" de circuitos "Y" (Fig. 7). El cuadro de certeza es el de la figura 6. Se utilizan todas las combinaciones: no existen combinaciones prohibidas.

A	B	C	Salidas
0	0	0	$P_0 = \bar{A}\bar{B}\bar{C}$
0	0	1	$P_1 = \bar{A}\bar{B}C$
0	1	0	$P_2 = \bar{A}B\bar{C}$
0	1	1	$P_3 = \bar{A}BC$
1	0	0	$P_4 = A\bar{B}\bar{C}$
1	0	1	$P_5 = A\bar{B}C$
1	1	0	$P_6 = AB\bar{C}$
1	1	1	$P_7 = ABC$

Fig. 6. *Cuadro de certeza de un decodificador de 2 x 3 entradas y 8 salidas*

Todos los circuitos "Y" tienen pues tres entradas, o sea, un total de 3 x 8 = 24 entradas sobre estos circuitos. En el caso de una decodificación de un número binario grande, el número de circuitos "Y", y las entradas a estos circuitos se hace muy grande. Para reducir el número de entradas de los circuitos "Y" se utilizan dos configuraciones:

– la configuración "en árbol",
– la configuración "equilibrada".

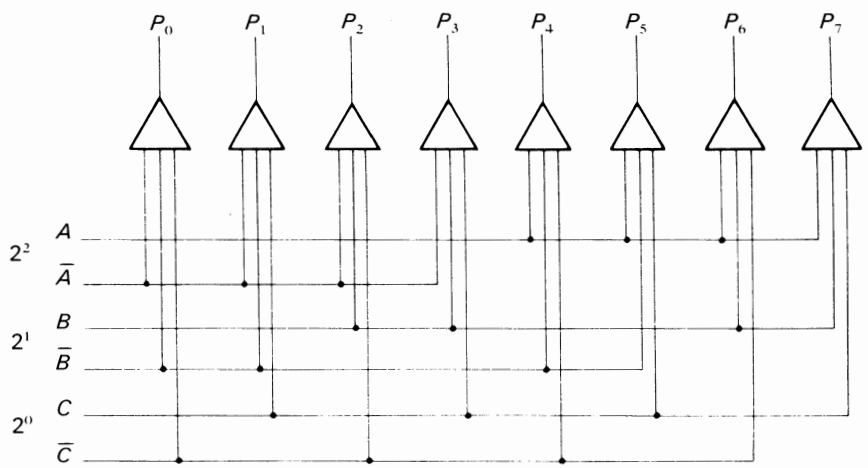


Fig. 7. *Matriz rectangular de un decodificador BCD ⇒ decimal*

En el caso de una matriz rectangular el número de circuitos "Y" es 2^n y siendo n el número de entradas sobre cada uno, el número de pares de variables de entrada de los circuitos "Y" es $n.2^n$. A cada una de las entradas de un circuito

"Y" corresponde un diodo de semiconductor. El coste de una matriz, se evalúa pues, en función del número total de entradas de los circuitos "Y". Para n = 3, tenemos $3.2^3 = 24$ entradas y para N = 4, $4.2^4 = 64$ entradas. Para disminuir el número de entradas, y por consiguiente el coste de una matriz de decodificador, se utiliza la configuración en árbol.

2.2.2. *Configuración en árbol*

Consideremos un decodificador de $n = 4$ entradas, tendremos $2^4 = 16$ salidas separadas, siendo sensibilizadas cada una de ellas separadamente. La realización de la configuración en árbol corresponde a la realización de dieciséis términos P en la forma siguiente:

$$
\begin{array}{llll}
P_0 = [(\overline{AB})\,\overline{C}]\,\overline{D} & P_5 = [(\overline{A}B)\,\overline{C}]\,D & P_{11} = [(A\overline{B})\,C]\,D \\
P_1 = [(\overline{AB})\,\overline{C}]\,D & P_6 = [(\overline{A}B)\,C]\,\overline{D} & P_{12} = [(AB)\,\overline{C}]\,\overline{D} \\
P_2 = [(\overline{AB})\,C]\,\overline{D} & P_7 = [(\overline{A}B)\,C]\,D & P_{13} = [(AB)\,\overline{C}]\,D \\
P_3 = [(\overline{AB})\,C]\,D & P_8 = [(A\overline{B})\,\overline{C}]\,\overline{D} & P_{14} = [(AB)\,C]\,\overline{D} \\
P_4 = [(\overline{A}B)\,\overline{C}]\,\overline{D} & P_9 = [(A\overline{B})\,\overline{C}]\,D & P_{15} = [(AB)\,C]\,D \\
 & P_{10} = [(A\overline{B})\,C]\,\overline{D} &
\end{array}
$$

Primeramente se realizan los términos de P correspondientes a las dos variables A y B. Seguidamente se agrupan estos de dos en dos y se asocian al término C para dar los términos de P correspondientes a las tres variables A,B,C. Finalmente estos términos se asocian a la variable D. El número de circuitos "Y" necesarios para realizar el decodificador es para $n = 2$, 2^2, para $n = 3$, $2^2 + 2^3$ y para $n = 4$, $2^2 + 2^3 + 2^4$. *Representando n el número de pares de variables de entrada.* Así, para cuatro pares de variables de entrada se necesitarán 28 circuitos "Y" de dos entradas, lo que hace un total de 56 entradas para todos los circuitos "Y" ($2^2 + 2^4 + 2^5$). De un modo general el número de entradas de todos los circuitos "Y" será:

$$N = \sum_{r=2}^{n} 2^{r+1} .$$

En el caso de una matriz rectangular tendremos $n.2^n = 4.16 = 64$ entradas, mientras que con la configuración en árbol se tienen $2^3 + 2^4 + 2^5 = 56$ entradas. La realización de la matriz correspondiente será más económica ya que a cada entrada de un circuito "Y" corresponde en general un diodo (Fig. 8).

Para facilitar la comparación entre varios tipos de decodificadores se ha establecido una lista de características comunes a todos los tipos que comprende, en función del número de variables a la entrada, el número de salidas, el número de circuitos "Y" del decodificador, el número de entradas comunes a cada circuito "Y", el número total de entradas y un factor de carga. Por factor de carga se entiende el mayor número de circuitos "Y" que tienen una entrada común.

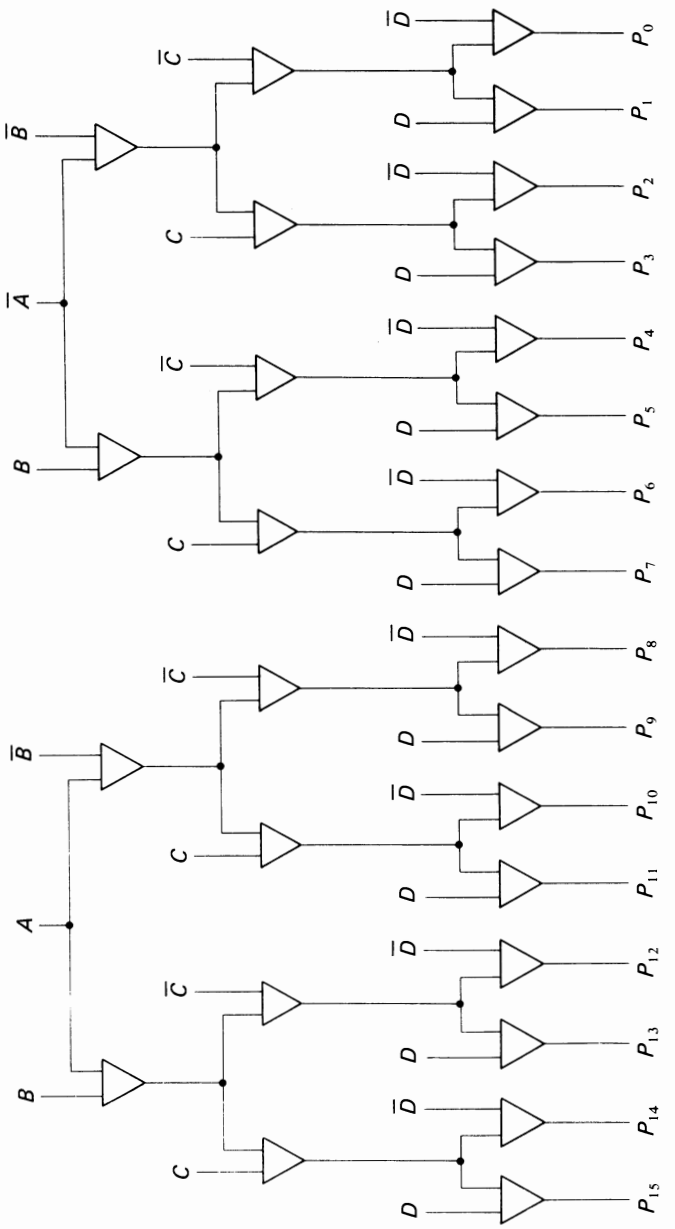


Fig. 8. *Decodificador BCD ⇒ decimal de cuatro variables. Configuración en árbol.*

Los cuadros 1 y 2 reflejan estas características para los decodificadores *P*, los decodificadores en árbol, en función del número de variables a la entrada. El decodificador *P* recibe también el nombre de *decodificador de "minterms"*, cada uno de los productos P_0 a P_7 representa un "minterm" de una función de tres variables.

Cuadro 1. Decodificador P (minterms).

Características	Dos variables *AB*	Tres variables *ABC*	Cuatro variables *ABCD*	Cinco variables *ABCDE*
Número de salidas	4	8	16	32
Número de circuitos "Y"	4	8	16	32
Número de entradas por circuitos "Y"	2	3	4	5
Número total de entradas	8	24	64	160
Factor de carga	2	4	8	16

Cuadro 2. Decodificador en árbol

Características	Dos variables *AB*	Tres variables *ABC*	Cuatro variables *ABCD*	Cinco variables *ABCDE*
Número de salidas	4	8	16	32
Número de circuitos "Y"	4	12	28	60
Número de entradas por circuitos "Y"	2	2	2	2
Número total de entradas	8	24	56	120
Factor de carga	2	4	8	16

Se notará que para los dos tipos de decodificadores el número de salidas y el factor de carga son idénticos. El número de circuitos "Y" es inferior para el decodificador *P*, pero el número de entradas de cada circuito "Y" aumenta con el número de variables a la entrada, lo que puede ser un inconveniente, mientras que ese número es constante e igual a 2 para el decodificador en árbol. El número total de entradas es menor para el decodificador en árbol a partir de cuatro variables a la entrada.

2.2.3. *Configuración equilibrada*

En este tipo de configuración se disminuye aún más el número de circuitos "Y" así como el número de entradas de estos circuitos. En estas configuraciones las variables de entradas se asocian de dos en dos. Cada circuito "Y" comprende dos entradas. En el caso de un decodificador en el que $n = 4$ (cuatro pares de variables de entrada) se realizarán todos los términos P del modo siguiente:

$$
\begin{aligned}
P_0 &= (\overline{A}\,\overline{B})\,(\overline{C}\,\overline{D}) & P_8 &= (A\overline{B})\,(\overline{C}\,\overline{D})\\
P_1 &= (\overline{A}\,\overline{B})\,(\overline{C}D) & P_9 &= (A\overline{B})\,(\overline{C}D)\\
P_2 &= (\overline{A}\,\overline{B})\,(C\overline{D}) & P_{10} &= (A\overline{B})\,(C\overline{D})\\
P_3 &= (\overline{A}\,\overline{B})\,(CD) & P_{11} &= (A\overline{B})\,(CD)\\
P_4 &= (\overline{A}B)\,(\overline{C}\,\overline{D}) & P_{12} &= (AB)\,(\overline{C}\,\overline{D})\\
P_5 &= (\overline{A}B)\,(\overline{C}D) & P_{13} &= (AB)\,(\overline{C}D)\\
P_6 &= (\overline{A}B)\,(C\overline{D}) & P_{14} &= (AB)\,(C\overline{D})\\
P_7 &= (\overline{A}B)\,(CD) & P_{15} &= (AB)\,(CD)\,.
\end{aligned}
$$

Se obtendrán de este modo 24 *circuitos* "Y" *de dos entradas, o sea, un total de* 48 entradas en lugar de 56 para la configuración en árbol y 64 para la matriz rectangular. Mientras más variables contenga el número a decodificar más se acentuará la diferencia. El *decodificador equilibrado es, pues, la solución más económica en el caso* en que se utilicen todas las salidas. El esquema de realización es el de la figura 9. Las configuraciones bajo forma de matrices de diodos pueden igualmente utilizarse para realizar otros tipos de circuitos. Veremos más adelante cómo una matriz de diodos permite realizar un adicionador completo. Entre los restantes dispositivos que se utilizan frecuentemente tienen gran importancia "los *contadores*" tanto por la cantidad como por sus funciones en el calculador.

El decodificador que acabamos de estudiar asocia las variables AB y CD entre ellas en dos grupos distintos, siendo éstos combinados entre ellos para obtener la decodificación final. Designaremos esta decodificación de cuatro variables por el símbolo siguiente: $ABCD$ (2, 2), lo que significa que la decodificación se efectúa por combinación de dos grupos de dos variables cada uno. Igualmente el mismo decodificador podrá realizarse combinando dos grupos de tres y una variables: $ABCD$ (3, 1). Es posible igualmente obtener un decodificador de cinco variables haciendo las combinaciones siguientes: $ABCD$ (3,2) o $ABCDE$ (2, 2, 1). La figura 9a da un ejemplo de la realización.

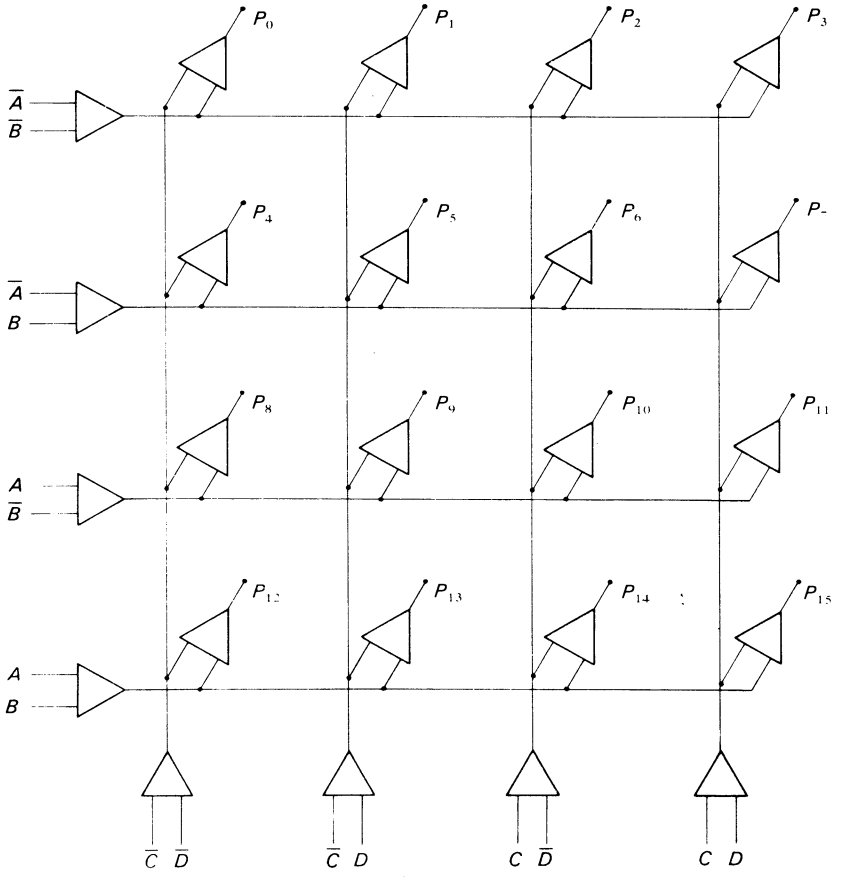


Fig. 9. *Decodificador BCD ⇒ decimal, configuración equilibrada.*

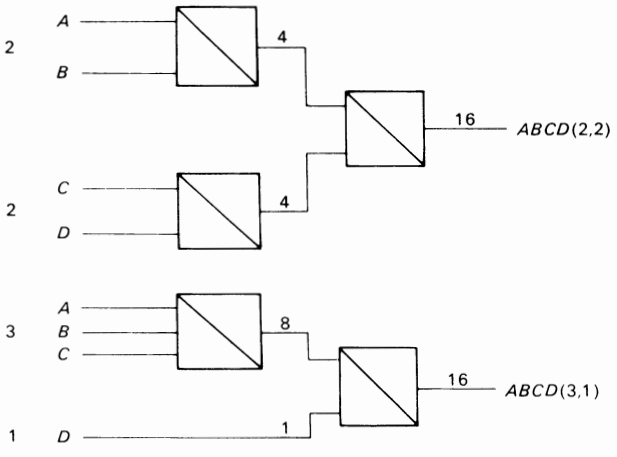


Fig. 9a. *Diferentes modos de realizar un decodificador equilibrado en cuatro variables.*

Los cuadros 3 y 4 reflejan las características de los decodificadores siguientes *ABC* (2, 2), *ABCD* (3, 1), *ABCDE* (3, 2), y *ABCDE* (2, 2, 1).

Cuadro 3. *Comparación de los decodificadores ABCD (2,2) y ABCD (3,1)*

Características	*ABCD* (2, 2)	*ABCD* (3, 1)
Número de salidas	16	16
Número de circuitos "Y"	24	24
Número total de entradas	48	56
Factor de carga	4	8

Cuadro 4. Comparación de los decodificadores ABCDE (3,2) y ABCDE (2,2,1)

Características	*ABCDE* (3, 2)	*ABCDE* (2, 2, 1)
Número de salidas	32	32
Número de circuitos "Y"	44	56
Número total de entradas	96	112
Factor de carga	8	4

El examen de los cuadros 3 y 4 nos da que el número de circuitos "Y" crece con la asimetría y el número de grupos de variables realizados para la

decodificación. Lo mismo ocurre para el número total de entradas. Por el contrario el factor de carga disminuye. Volviendo sobre los cuadros 1 y 2 se observará que el decodificador equilibrado es el que tiene un número más pequeño total de entradas.

2.3 Caso de códigos incompletos

Si el código utilizado es incompleto como es el caso general, ya que son valores decimales lo que hay que decodificar con mayor frecuencia, se suprimen las combinaciones que no existen (Fig. 9b). Conservándose de este modo la estructura equilibrada.

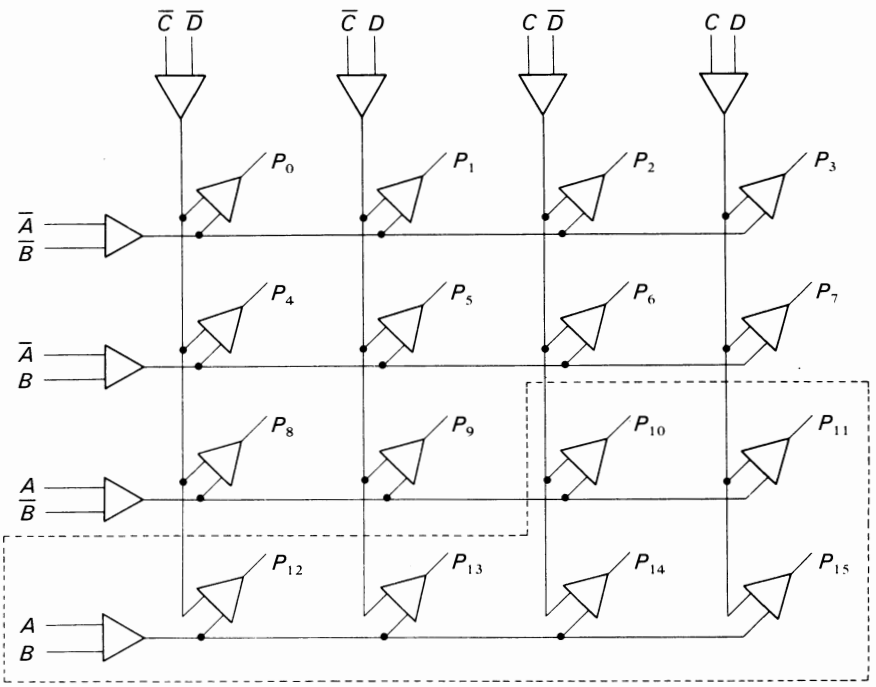


Fig. 9b. *Decodificador binario-decimal equilibrado. En punteado las condiciones prohibidas eliminadas del circuito*

Un decodificador algo más económico que el anterior se obtiene realizando un decodificador que combine las tres variables *ABC* entre ellas antes de asociarlas con la variable *D* (Fig. 9c). Siendo eliminadas del circuito las combinaciones prohibidas.

Se dispone de 15 circuitos "Y" en lugar de los 17 del circuito precedente.

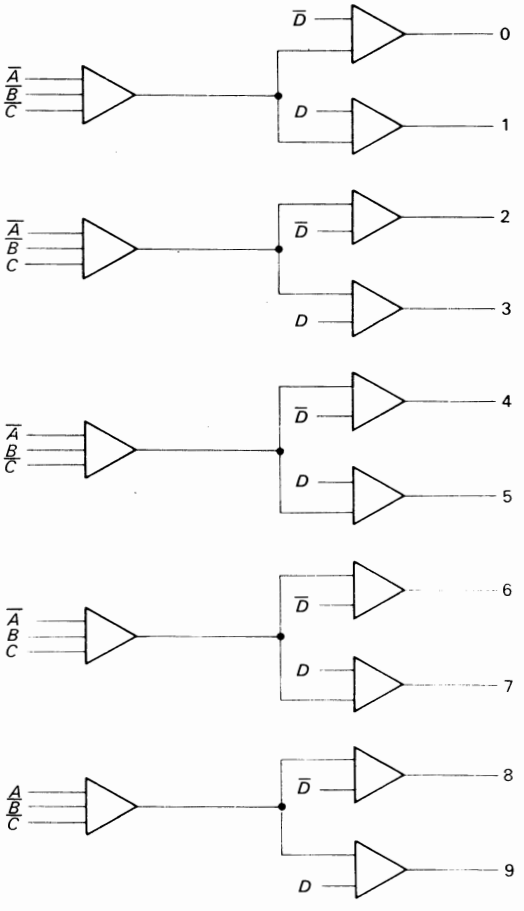


Fig. 9c. *Decodificador binario decimal, segunda versión*

El número total de entradas es de 35 en vez de 34 para el decodificador equilibrado. Este decodificador se utiliza frecuentemente.

2.4 Los transcodificadores

Se llama transcodificador a un sistema que permite pasar de un código a otro. Tomemos como ejemplo el código binario reflejado + 3 que se desea convertir en código 1.2.4.8. Existen dos posibilidades:

1. Decodificar uno de los códigos en el sistema decimal y codificar el resultado en el segundo código.

2. Transposición directa de un código a otro por medio del álgebra de Boole.

Examinaremos sucesivamente los dos casos. El código binario reflejado + 3 es el código reflejado desplazado de tres posiciones para que el paso de 9 a 0 solamente afecte al cambio de un solo dígito del número codificado. Este código es el siguiente:

	A B C D	
×	0 0 0 0	
×	0 0 0 1	
×	0 0 1 1	
0	0 0 1 0	$0 = \overline{A}\overline{B}C\overline{D}$
1	0 1 1 0	$1 = \overline{A}BC\overline{D}$
2	0 1 1 1	$2 = \overline{A}BCD$
3	0 1 0 1	$3 = \overline{A}B\overline{C}D$
4	0 1 0 0	$4 = \overline{A}B\overline{C}\overline{D}$
5	1 1 0 0	$5 = AB\overline{C}\overline{D}$
6	1 1 0 1	$6 = AB\overline{C}D$
7	1 1 1 1	$7 = ABCD$
8	1 1 1 0	$8 = ABC\overline{D}$
9	1 0 1 0	$9 = A\overline{B}C\overline{D}$
×	1 0 1 1	
×	1 0 0 1	
×	1 0 0 0	

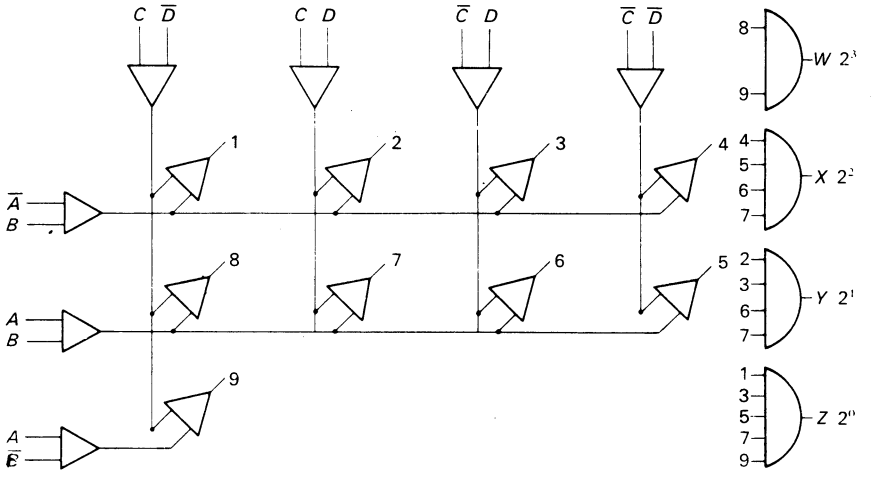


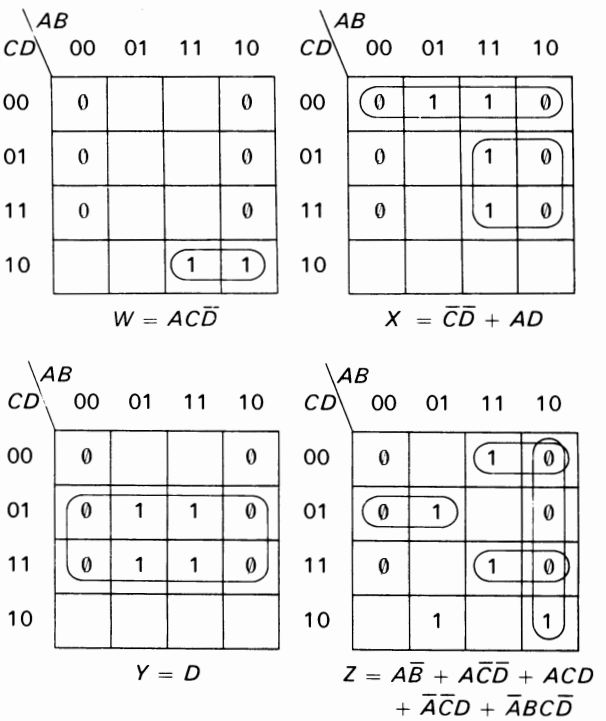
Fig. 9d. *Paso del código binario reflejado + 3 al código 1248*

Las cruces señalan las condiciones prohibidas. Si se desea realizar un decodificador totalmente inmunizado contra estas condiciones prohibidas, se les eliminará del circuito de decodificación. El valor 0 no será decodificado ya que no interviene en el circuito de decodificación siguiente.

La transposición de un código a otro se efectúa según el esquema de la figura 9d.

El segundo método consiste en realizar un circuito único que hace la transcodificación. A cada valor del código binario reflejado + 3 se le atribuye el valor correspondiente en el código 1.2.4.8. Atribuyéndose a las condiciones prohibidas el valor Ø. Estableciendo los diagramas de Karnaugh para *W, X, Y, Z,* obtenemos:

	A B C D	*W X Y Z*
X	0 0 0 0	Ø Ø Ø Ø
X	0 0 0 1	Ø Ø Ø Ø
X	0 0 1 1	Ø Ø Ø Ø
0	0 0 1 0	0 0 0 0
1	0 1 1 0	0 0 0 1
2	0 1 1 1	0 0 1 0
3	0 1 0 1	0 0 1 1
4	0 1 0 0	0 1 0 0
5	1 1 0 0	0 1 0 1
6	1 1 0 1	0 1 1 0
7	1 1 1 1	0 1 1 1
8	1 1 1 0	1 0 0 0
9	1 0 1 0	1 0 0 1
X	1 0 1 1	Ø Ø Ø Ø
X	1 0 0 1	Ø Ø Ø Ø
X	1 0 0 0	Ø Ø Ø Ø



La figura 9e refleja el esquema de la realización.

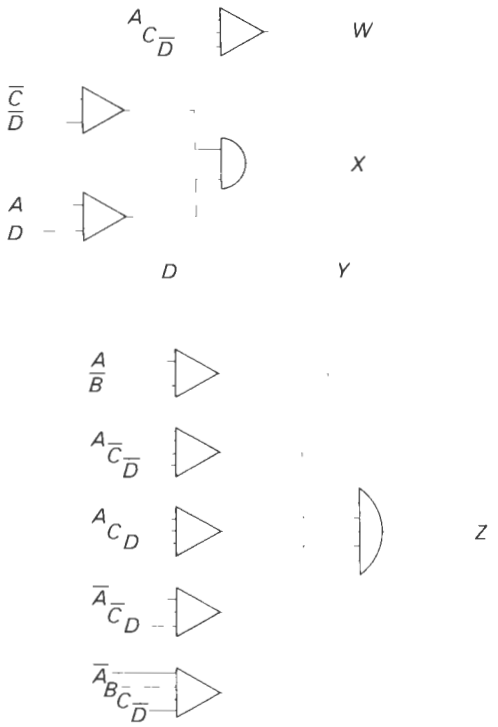


Fig. 9e. *Paso del código binario reflejado + 3 al código BCD*

La comparación entre los dos métodos indica que el segundo es mucho más económico: 10 circuitos "Y" y "O" contra 16 circuitos "Y" y 3 circuitos "O". El número total de entradas para el primer método es de 48 mientras que en el segundo son 29. Se economiza casi el 50 $^{o}/_{o}$ con el segundo método. Esto es debido a que se han incorporado las condiciones prohibidas en el cálculo para simplificar el circuito, mientras que en el primer método éstas han sido eliminadas al iniciar el cálculo. El primer circuito es completamente insensible a las condiciones prohibidas si éstas se produjeran, lo que no ocurrirá en el segundo circuito en el que la aparición de una condición prohibida en la entrada da un resultado lógico no deseable en la salida. Es pues necesario, en el segundo caso, asegurarse de que las condiciones prohibidas no se producirán nunca o que si estas aparecen no tengan ninguna influencia sobre la continuación de las operaciones lógicas.

3. Los contadores

Se llama contador a un dispositivo que presenta ciertos estados bajo excitación exterior. Los contadores son circuitos secuenciales que responden a una secuencia de impulsos que reciben el nombre de impulsos de avance. Los estados del contador pueden ser cualesquiera o representar la suma acumulada de los impulsos de avance. En este caso se cuentan los impulsos de entrada.

De un modo general se emplea la denominación contador para designar un dispositivo cuyos sucesivos estados representan un número binario creciente o decreciente en función de un impulso de avance. Los contadores se realizan con las básculas *T*, *R-S*, *R-S-T* o *J-K*.

3.1 Determinación analítica de los contadores

Para determinar un contador se procederá del modo siguiente:

– Definir *la capacidad máxima* del contador, es decir el mayor número binario que el contador es capaz de representar. Determinándose así el número de etapas. Con n etapas se cuenta hasta 2^n-1.

– Elegir el código a utilizar (código binario puro, código + 3, código Gray, etc.).

– Establecer el estado sucesivo de las básculas utilizadas a cada impulso de avance en función del código elegido.

– Por medio de los cuadros de certeza de las básculas utilizadas (estudiadas en el tomo I), determinar cuáles son los valores de las entradas de cada báscula que definen el estado del contador a cada impulso de avance.

– Seguidamente llevar los valores de las entradas (*R*, *S*, *J*, *K*, *T*) a un diagrama de Karnaugh y simplificar.

– Realizar el circuito.

– Verificar el funcionamiento.

Utilizaremos este método para la determinación de varios tipos de contadores.

3.2 Contador binario puro. Utilización de las básculas R-S

En este tipo de contador el código utilizado es el binario puro. Para determinar este contador se comienza por estudiar la configuración de un contador de capacidad máxima 7 utilizando las básculas *R-S*. Por iteración se determina seguidamente la configuración de un contador de capacidad mayor. Para contar hasta 7 según el código binario puro, es necesario disponer de tres etapas ($2^3 = 8$). Denominaremos a estas etapas *A*, *B*, *C*, representadas cada una por una báscula *R-S*. Las respectivas entradas de cada una de las básculas se designan por R_A, S_A, R_B, S_B, R_C, S_C.

El cuadro de certeza de la báscula *R-S* ha sido estudiado anteriormente. Lo reproduciremos a continuación (Fig. 10). Este cuadro de certeza ha sido modificado para hacer aparecer las condiciones que son necesarias a R y S para que la báscula cambie de estado o conserve su estado anterior. Tendremos pues cuatro condiciones según que el estado anterior sea $Q_t = 0$ o $Q_t = 1$.

Q_t	R	S	Q_{t+1}
0	0	0	0
0	0	1	1
0	1	0	0
1	0	0	1
1	0	1	1
1	1	0	0

Cuadro 10. Cuadro de certeza de la báscula R-S

Este es el cuadro que utilizaremos para determinar los diferentes tipos de contadores que utilizan las básculas *R-S* (Fig. 11). En este cuadro $\emptyset$ es un valor opcional que podrá tomar indiferentemente el valor 0 ó 1. Se deberá escoger el valor conveniente para obtener la mayor simplificación. Seguidamente se establece el cuadro de certeza del contador, es decir, los estados sucesivos de cada báscula así como los estados a imponer a las entradas R y S. La figura 12 representa el cuadro de certeza para el código binario puro.

Q_t	R	S	Q_{t+1}
0	$\emptyset$	0	0
0	0	1	1
1	1	0	0
1	0	$\emptyset$	1

Fig. 11. *Cuadro de certeza modificado de la báscula R-S*

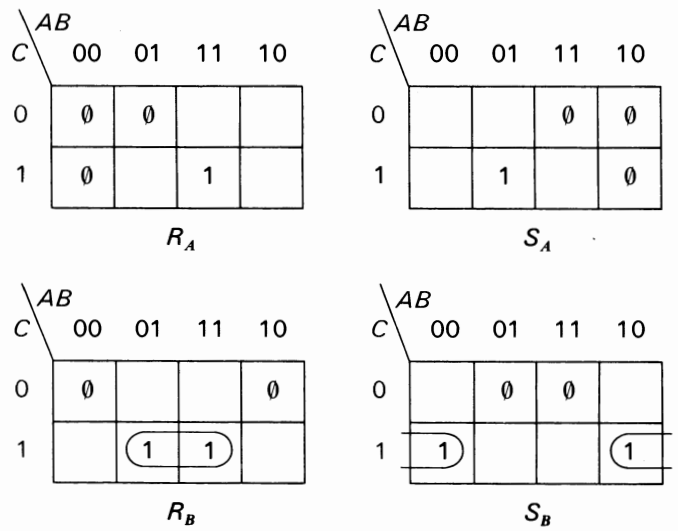
El contador cuenta de 0 a 7 volviendo a 0 con el octavo impulso de avance para tomar de nuevo su ciclo inicial: es lo que se llama el ciclo del contador. Refiriéndonos a la figura 11, se obtienen los valores de R_A, S_A, R_B S_B, R_C, y S_C de cada báscula A. B y C en función de su estado anterior y de su nuevo estado. Así, la báscula A conserva su estado 0 en el primer impulso de avance. Según el cuadro de certeza de la figura 11 tendremos $R_A = \emptyset$, $S_A = 0$.

Decimal	Avc	A	B	C	R_A	S_A	R_B	S_B	R_C	S_C
0	1	0	0	0	∅	0	∅	0	0	1
1	1	0	0	1	∅	0	0	1	1	0
2	1	0	1	0	∅	0	0	∅	0	1
3	1	0	1	1	0	1	1	0	1	0
4	1	1	0	0	0	∅	∅	0	0	1
5	1	1	0	1	0	∅	0	1	1	0
6	1	1	1	0	0	∅	0	∅	0	1
7	1	1	1	1	1	0	1	0	1	0
0	1	0	0	0						

Fig. 12. *Cuadro de certeza del contador binario puro*

Al cuarto impulso de avance la báscula A pasa del estado 0 al estado 1, lo que corresponde a $R_A = 0$ y $S_A = 1$. Estas condiciones deben ser aplicadas a R_A y S_A antes del impulso de avance. En efecto, es necesario aplicar estas condiciones a R_A y S_A, y al producirse el impulso de avance siguiente la báscula cambiará o conservará su estado. Las flechas de la figura 12 señalan esta eventualidad. Seguidamente se procede del mismo modo para las básculas B y C. Obteniéndose así los valores de R_B, S_B, R_C y S_C que se han señalado seguidamente en un diagrama de Karnaugh para simplificar (Fig. 12 bis).

Según el cuadro de la figura 12 obtendremos los diagramas de Karnaugh para las entradas R y S de las tres básculas en función de los diferentes estados del contador (Fig. 12 bis).



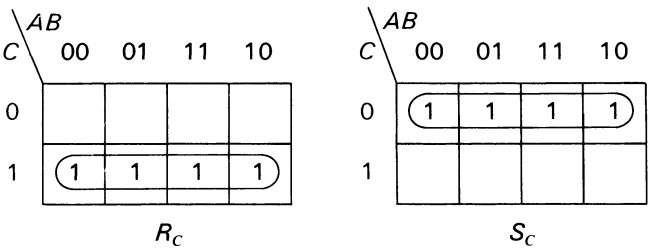


Fig. 12bis. *Diagramas de Karnaugh de las entradas R-S*

La lectura del diagrama da (Fig. 12 bis).

$$R_A = (ABC)\ \text{Avc} \qquad R_B = (BC)\ \text{Avc} \qquad R_C = (C)\ \text{Avc}\ ,$$
$$S_A = (\overline{A}BC)\ \text{Avc} \qquad S_B = (\overline{B}C)\ \text{Avc} \qquad S_C = (\overline{C})\ \text{Avc}\ .$$

Según el cuadro de la figura 12 el cambio de estado de cada báscula se efectúa coincidiendo con el avance.

Obsérvese la ley de formación de los términos R y S. Para un contador de cuatro básculas, es decir que cuente hasta $2^4 - 1 = 15$, sea de 0 a 15, tendremos:

$$R_A = (ABCD)\ \text{Avc} \quad R_B = (BCD)\ \text{Avc} \quad R_C = (CD)\ \text{Avc} \quad R_D = (D)\ \text{Avc}\ ,$$
$$S_A = (\overline{A}BCD)\ \text{Avc} \quad S_B = (\overline{B}CD)\ \text{Avc} \quad S_C = (\overline{C}D)\ \text{Avc} \quad S_D = (\overline{D})\ \text{Avc}\ .$$

Del mismo modo se procederá para un contador de cinco básculas. El esquema de realización de un contador de tres básculas está señalado en la figura 13.

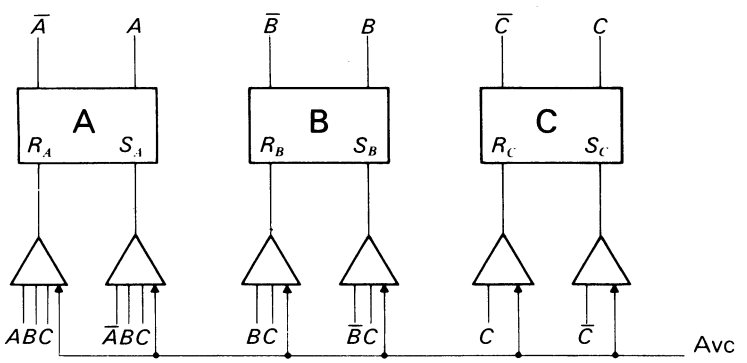


Fig. 13. *Contador binario de capacidad máxima 7*

3.2.1 *Contador que refleja la sucesión de números impares*

En este ejemplo se desea realizar un contador con 7 como capacidad máxima, compuesto, pues,de tres básculas *R-S* que reflejan por cada impulso de avance la sucesión de los números impares. El ciclo del contador será 0-1-3-5-7-0. Los valores binarios que corresponden a los números decimales 2, 4, 6 no son señalados. Son las *condiciones prohibidas.* Para estas condiciones se podrá aplicar indiferentemente 0 ó 1 sobre las entradas *R* y *S*, es decir Ø. El cuadro de certeza del contador se da en la figura 13 bis.

	Decimal	Avc	*A*	*B*	*C*	R_A	S_A	R_B	S_B	R_C	S_C
	0	1	0	0	0	Ø	0	Ø	0	0	1
	1	1	0	0	1	Ø	0	0	1	0	Ø
	3	1	0	1	1	0	1	1	0	0	Ø
	5	1	1	0	1	0	Ø	0	1	0	Ø
	7	1	1	1	1	1	0	1	0	1	0
	0	1	0	0	0						
condiciones prohibidas	2	1	0	1	0	Ø	Ø	Ø	Ø	Ø	Ø
	4	1	1	0	0	Ø	Ø	Ø	Ø	Ø	Ø
	6	1	1	1	0	Ø	Ø	Ø	Ø	Ø	Ø

Fig. 13bis. *Cuadro de certeza del contador que da la sucesión de números impares de 0 a 7*

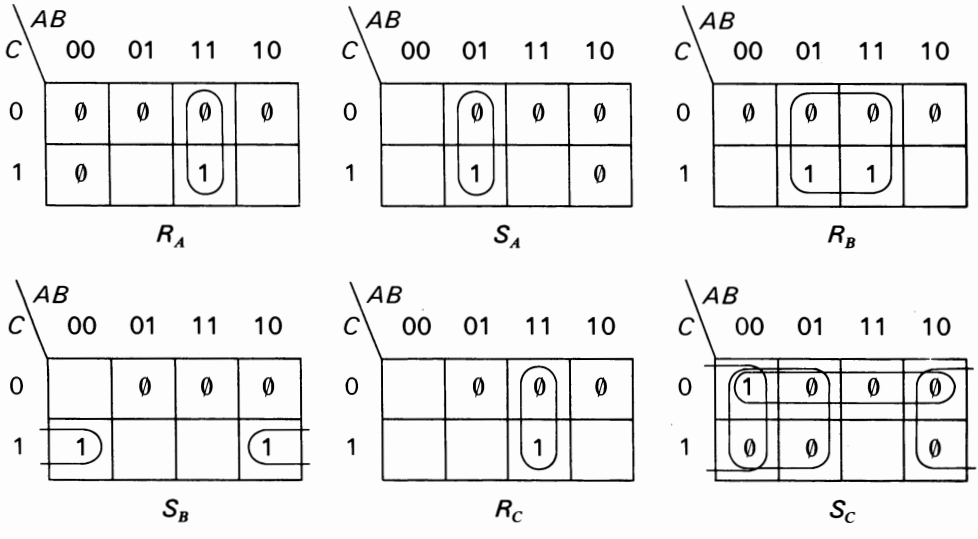


Fig. 13ter. *Diagrama de Karnaugh de las entradas R y S del contador de la figura 14*

Los valores de R y S que corresponden a las condiciones prohibidas son simplificados en los diagramas de Karnaugh (Fig. 13 ter.).

La lectura de estos diagramas da coincidiendo con el vance:

$$S_A = (\overline{A}B)\ \text{Avc} \qquad S_B = (\overline{B}C)\ \text{Avc} \qquad S_C = \begin{cases} (\overline{C})\ \text{Avc} \\ (\overline{A})\ \text{Avc} \\ (\overline{B})\ \text{Avc} \end{cases}$$

$$R_A = (AB)\ \text{Avc} \qquad R_B = (B)\ \text{Avc} \qquad R_C = (AB)\ \text{Avc}\ .$$

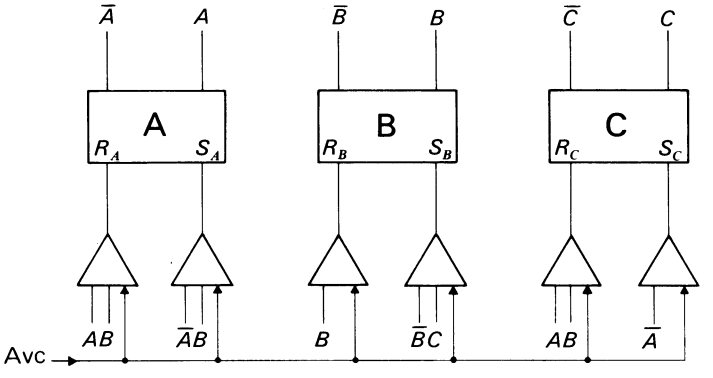


Fig. 14. *Contador de capacidad máxima 7 para la sucesión de los números impares 0-1-3-5-7-0*

Obsérvense los tres bucles de simplificación de S_C. Eligiendo para S_C la solución $S_C = (\overline{A})$ Avc, el esquema del contador es el de la figura 14.

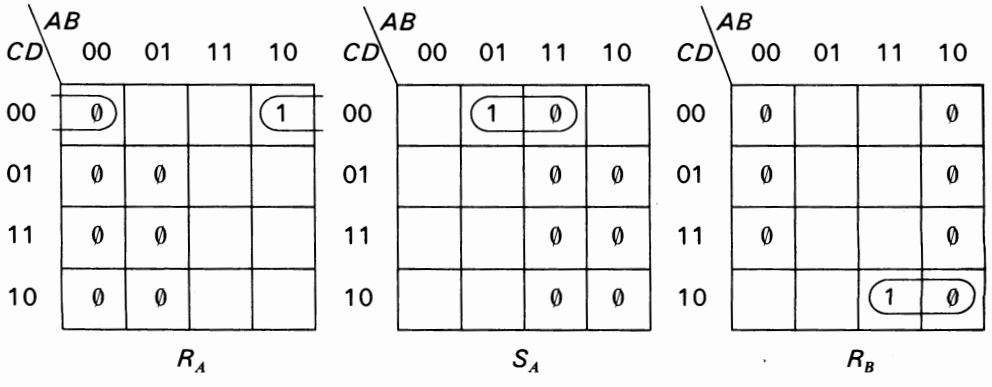
3.2.2 *Contador que utiliza el código de Gray o binario reflejado*

Estudiaremos en este párrafo un contador con 16 de capacidad máxima, contando de 0 a 15 según el código Gray. Estará compuesto de cuatro básculas A, B, C y D.

Estas básculas son del tipo RS. El código Gray o código binario reflejado ha sido estudiado en el capítulo 3 del tomo I. Este código tiene la particularidad de que es un solo bit el que cambia, cuando se pasa de un valor decimal al siguiente. Es frecuentemente utilizado en los codificadores de ángulos. La figura 15 representa el cuadro de certeza de este tipo de contador así como las condiciones a aplicar a las entradas R y S. La figura 16 representa los diagramas de Karnaugh relativos a las entradas R y Sde cada una de las cuatro básculas A, B, C y D.

Decimal	Avc	A	B	C	D	R_A	S_A	R_B	S_B	R_C	S_C	R_D	S_D
0	1	0	0	0	0	∅	0	∅	0	∅	0	0	1
1	1	0	0	0	1	∅	0	∅	0	0	1	0	∅
2	1	0	0	1	1	∅	0	∅	0	0	∅	1	0
3	1	0	0	1	0	∅	0	0	1	0	∅	∅	0
4	1	0	1	1	0	∅	0	0	∅	0	∅	0	1
5	1	0	1	1	1	∅	0	0	∅	1	0	0	∅
6	1	0	1	0	1	∅	0	0	∅	∅	0	1	0
7	1	0	1	0	0	0	1	0	∅	∅	0	∅	0
8	1	1	1	0	0	0	∅	0	∅	∅	0	0	1
9	1	1	1	0	1	0	∅	0	∅	0	1	0	∅
10	1	1	1	1	1	0	∅	0	∅	0	∅	1	0
11	1	1	1	1	0	0	∅	1	0	0	∅	∅	0
12	1	1	0	1	0	0	∅	∅	0	0	∅	0	1
13	1	1	0	1	1	0	∅	∅	0	1	0	0	∅
14	1	1	0	0	1	0	∅	∅	0	∅	0	1	0
15	1	1	0	0	0	1	0	∅	0	∅	0	∅	0
0	1	0	0	0	0								

Fig. 15. *Cuadro de certeza del contador que utiliza el código Gray*



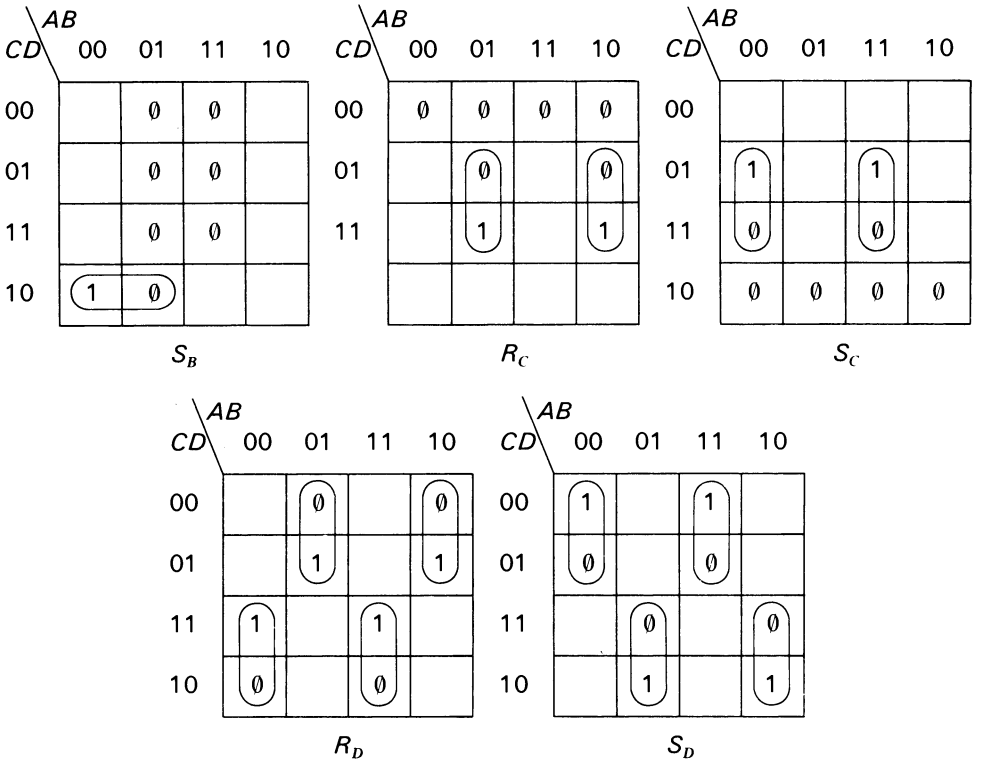


Fig. 16. *Diagramas de Karnaugh de las entradas del contador que utiliza el código Gray*

La lectura de los diagramas de Karnaugh da, coincidiendo con el avance, las ecuaciones lógicas siguientes:

$$R_A = (\overline{B}C\overline{D})\,\text{Avc} \qquad R_B = (AC\overline{D})\,\text{Avc}\,,$$
$$S_A = (B\overline{C}\overline{D})\,\text{Avc} \qquad S_B = (\overline{A}C\overline{D})\,\text{Avc}\,,$$
$$R_C = D(\overline{A}B + A\overline{B})\,\text{Avc} = D(A \oplus B)\,\text{Avc}\,,$$
$$S_C = D(\overline{A}\overline{B} + AB)\,\text{Avc} = D(A \odot B)\,\text{Avc} = D\overline{(A \oplus B)}\,\text{Avc}\,,$$
$$R_D = [C(\overline{A}\overline{B} + AB) + \overline{C}(\overline{A}B + A\overline{B})]\,\text{Avc}\,,$$
$$S_D = [\overline{C}(\overline{A}\overline{B} + AB) + C(\overline{A}B + A\overline{B})]\,\text{Avc}\,,$$

que pueden escribirse:

$$R_D = [C(A \odot B) + \overline{C}(A \oplus B)]\,\text{Avc}\,,$$
$$S_D = [\overline{C}(A \odot B) + C(A \oplus B)]\,\text{Avc}\,.$$

o lo que es lo mismo:

$$R_D = [C \oplus (A \oplus B)]\,\text{Avc}\,,$$
$$S_D = [\overline{C \oplus (A \oplus B)}]\,\text{Avc}\,.$$

Recapitulando los resultados:

$$R_A = (\overline{B}\overline{C}\overline{D})\,\text{Avc} \qquad R_B = (A\overline{C}\overline{D})\,\text{Avc}$$
$$S_A = (B\overline{C}\overline{D})\,\text{Avc} \qquad S_B = (\overline{A}\overline{C}\overline{D})\,\text{Avc}$$
$$R_C = D(A \oplus B)\,\text{Avc} \qquad R_D = [C \oplus (A \oplus B)]\,\text{Avc}$$
$$S_C = D(\overline{A \oplus B})\,\text{Avc} \qquad S_D = [\overline{C \oplus (A \oplus B)}]\,\text{Avc}\,.$$

El esquema de realización es el representado en la figura 17.

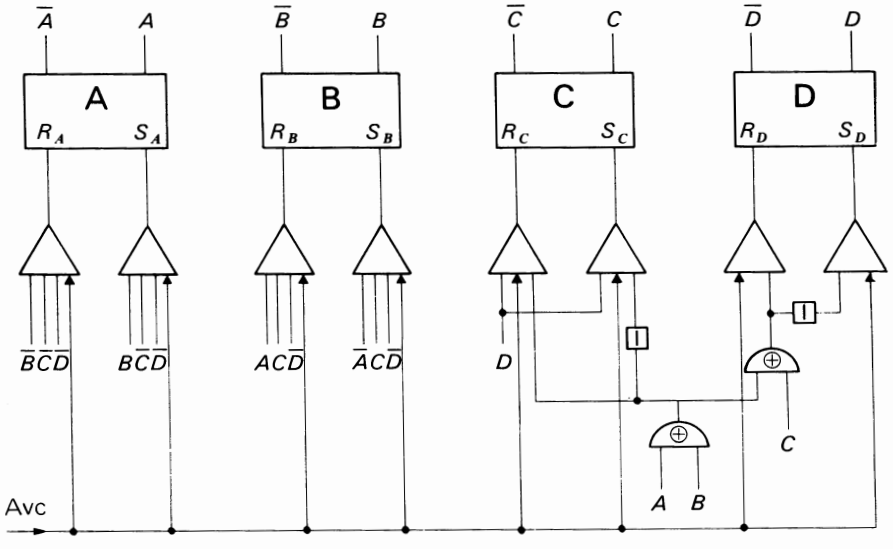


Fig. 17. *Contador de capacidad máxima 15 que utiliza el código Gray*

3.3. Contadores que utilizan las básculas R-S-T

La configuración de los contadores se puede simplificar a veces utilizando básculas *R-S-T*. Establecimos anteriormente el cuadro de certeza de la báscula *R-S-T* (ver Cap. 1 y 2, tomo I), no obstante lo reproducimos a continuación para una exposición más cómoda (Fig. 18).

Q_t	R	S	T	Q_{t+1}
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	1	0	0	0
1	0	0	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	0

Fig. 18. *Cuadro de certeza de la báscula R-S-T.*

Este cuadro de certezase puede modificar a fin de representar los cambios de estado entre Q_t y $Q_t + 1$ (Fig. 19). En este cuadro se consideran las condiciones a aplicar a la báscula para que ésta conserve su estado anterior o cambie de estado, para el estado anterior $Q_t = 0$ y $Q_t = 1$. Los valores $\emptyset$ son "*valores opcionales*", pudiendo ser indiferentemente 0 ó 1. Los ① señalados con un círculo son "*alternativas*". Si se elige uno de ellos, el que está situado sobre la misma línea debe ser nulo en virtud del principio de que no pueden existir dos 1 simultáneamente sobre dos entradas de la báscula *R-S-T*.

Q_t	R	S	T	Q_{t+1}
0	$\emptyset$	0	0	0
0	0	①	①	1
1	0	$\emptyset$	0	1
1	①	0	①	0

Fig. 19. *Cuadro de certeza simplificado de la báscula R-S-T-$\emptyset$ = es un valor opcional (0 ó 1),* ① *una alternativa. (No puede existir simultáneamente dos* ① *sobre la misma línea)*

Examinaremos a continuación algunas realizaciones de contadores que utilizan las básculas *R-S-T*.

3.3.1 *Contador para los valores impares con 7 de capacidad máxima que utiliza básculas R-S-T*

Este contador ya fue realizado con las básculas *R-S*. Examinaremos cuál será la nueva configuración del circuito utilizando las básculas *R-S-T*. El cuadro de

certeza del contador está representado en la figura 20. Obsérvese que la existencia de las condiciones prohibidas corresponde a los valores decimales 2, 4 y 6.

Decimal	Avc	A	B	C	R_A	S_A	T_A	R_B	S_B	T_B	R_C	S_C	T_C
0	1	0	0	0	∅	0	0	∅	0	0	0	①	①
1	1	0	0	1	∅	0	0	0	①	①	0	∅	0
3	1	0	1	1	0	①	①	①	0	①	0	∅	0
5	1	1	0	1	0	∅	0	0	①	①	0	∅	0
7	1	1	1	1	①	0	①	①	0	①	①	0	①
0	1	0	0	0									
Condiciones	1	0	1	0	∅	∅	∅	∅	∅	∅	∅	∅	∅
prohibidas	1	1	0	0	∅	∅	∅	∅	∅	∅	∅	∅	∅
(2, 4, 6)	1	1	1	0	∅	∅	∅	∅	∅	∅	∅	∅	∅

Fig. 20. *Cuadro de certeza del contador (ver texto).*

La figura 21 representa los diagramas de Karnaugh relativos a las entradas. De todos modos, antes de confeccionar estos diagramas deberán buscarse siempre sobre el cuadro de certeza las soluciones más simples. Se observará, sobre la figura 20, que existen varias soluciones según los valores dados a ∅ y los valores de ① elegidos.

3.3.2 *Primera solución*

La solución más simple consiste en conservar los ① en cada columna T. Anulando los ① en las columnas correspondientes de R y S. Además, debe observarse que eligiendo $\emptyset = 0$, todas las columnas R y S son nulas. Quedan únicamente las columnas T. *Lo que equivale a realizar un contador solamente con las básculas T.*

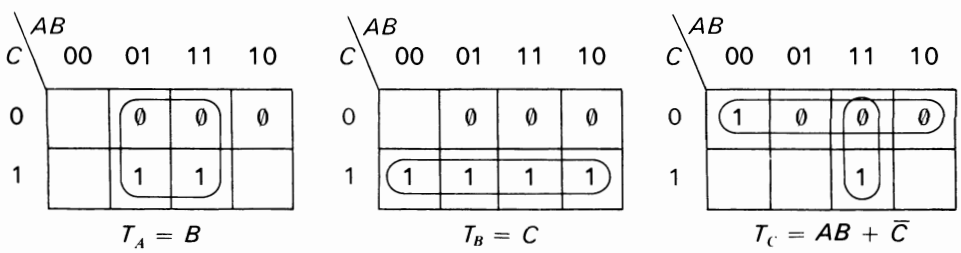


Fig. 21. *Diagramas de Karnaugh del contador de capacidad máxima para la sucesión de los números binarios 0-1-3-5-7-0*

La lectura de los diagramas y lo que acabamos de ver, da

$$R_A = 0 \qquad T_B = 0 \qquad R_C = 0$$

$$S_A = 0 \qquad S_B = 0 \qquad S_C = 0$$

$$T_A = (B)\,\text{Avc} \qquad T_B = (C)\,\text{Avc} \qquad T_C = (AB + \overline{C})\,\text{Avc}\ .$$

El esquema del contador está representado en la figura 22.

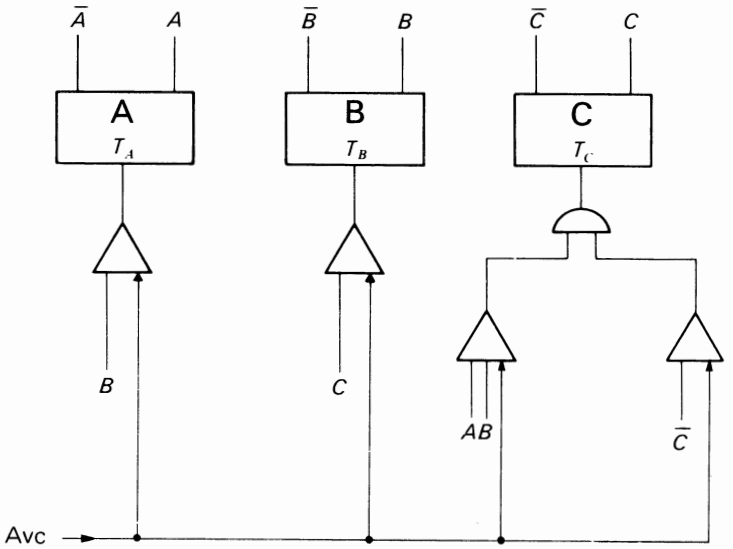


Fig. 22. *Contador de capacidad máxima 7 para la sucesión de los números impares (primera versión básculas T)*

3.3.3 *Segunda solución*

Se anulan los ① en T_B y se conservan los ① en las columnas S_B y R_B. Igualmente se toma $\emptyset = 0$ en todas las columnas. Las únicas condiciones que cambian son las relativas a S_B, R_B y T_B. Obtenemos $T_B = 0$ y R_B y S_B son dadas por los diagramas de Karnaugh en la figura 23, finalmente tenemos:

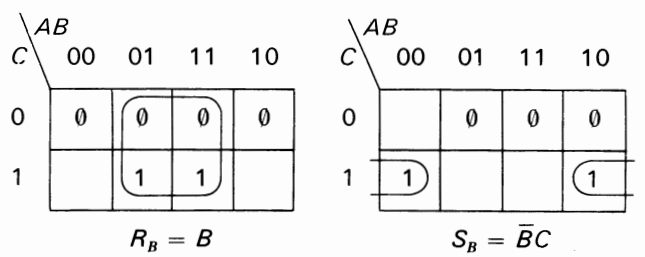


Fig. 23. *Diagramas de Karnaugh de la segunda solución*

$$R_A = 0 \qquad R_B = (B)\,\text{Avc} \qquad R_C = 0$$
$$S_A = 0 \qquad S_B = (\overline{B}C)\,\text{Avc} \qquad S_C = 0$$
$$T_A = (B)\,\text{Avc} \qquad T_B = 0 \qquad T_C = (AB + \overline{C})\,\text{Avc}\,.$$

El esquema del contador es el representado en la figura 24. Esta solución es menos económica que la anterior.

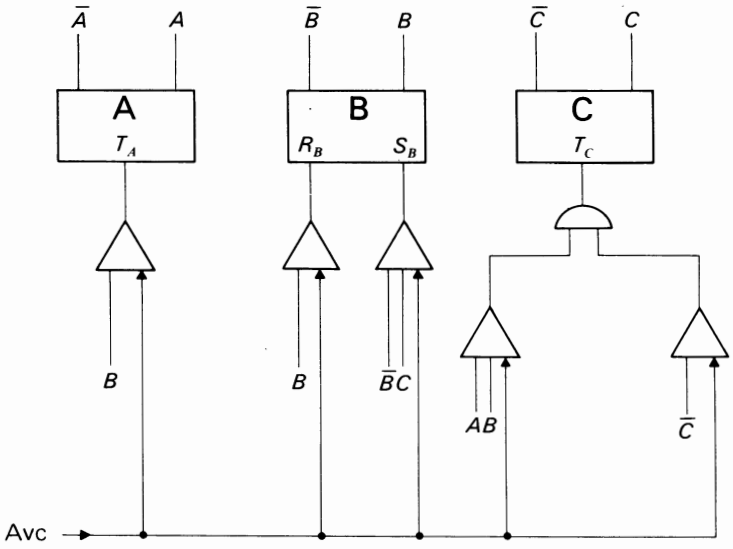


Fig. 24. *Esquema equivalente del contador (segunda versión)*

3.3.4 *Tercera solución*

Se conservan los ① en las columnas T_A y T_B y se les anula en las columnas T_C. ∅ permanece nulo. Para A y B se dan las mismas condiciones que en la primera versión. Los diagramas de Karnaugh de R_C y S_C se representan en la figura 25. Con estas condiciones se obtiene:

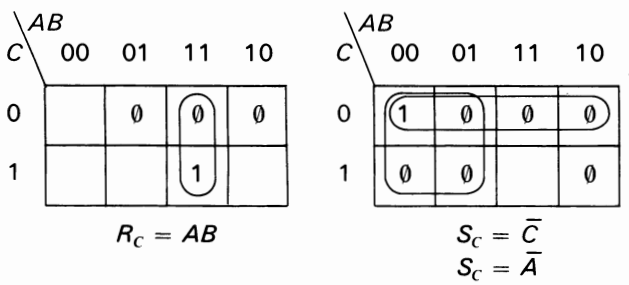


Fig. 25. *Diagramas de Karnaugh del contador (tercera versión)*

$R_A = 0$ $R_B = 0$ $R_C = (AB)$ Avc

$S_A = 0$ $S_B = 0$ $S_C = (\overline{C})$ Avc

$T_A = (B)$ Avc $T_B = (C)$ Avc $T_C = 0$.

El esquema de su realización es el representado en la figura 26, tomando para S_C el valor $S_C = \overline{C}$.

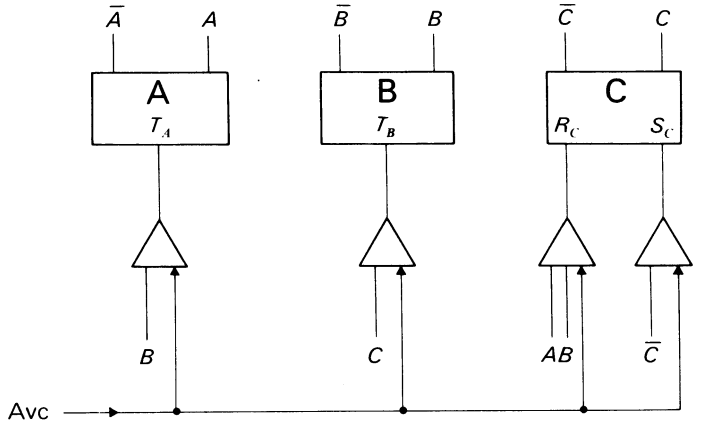


Fig. 26. *Esquema del contador tercera versión*

3.4 Contadores que utilizan las básculas T. Contador binario puro

En este párrafo veremos el caso de los contadores que utilizan las básculas *T*. Consideremos como ejemplo el caso del contador binario puro con 7 de capacidad máxima. El cuadro de certeza de la báscula *T* ha sido estudiado anteriormente. Lo reproducimos en la figura 27 para mayor facilidad. El contador comprende tres etapas y se pone a cero después de dar el valor binario 111 que corresponde al 7 decimal. El cuadro de certeza del contador se representa en la figura 28. Seguidamente se llevan los valores T_A, T_B y T_C a un diagrama de Karnaugh para simplificación.

Q_t	T	Q_{t+1}
0	0	0
0	1	1
1	0	1
1	1	0

Fig. 27. *Cuadro de certeza de la báscula T*

Decimal	Avc	A	B	C	T_A	T_B	T_C
0	1	0	0	0	0	0	1
1	1	0	0	1	0	1	1
2	1	0	1	0	0	0	1
3	1	0	1	1	1	1	1
4	1	1	0	0	0	0	1
5	1	1	0	1	0	1	1
6	1	1	1	0	0	0	1
7	1	1	1	1	1	1	1
0	1	0	0	0			

Fig. 28. *Cuadro de certeza del contador binario de básculas T*

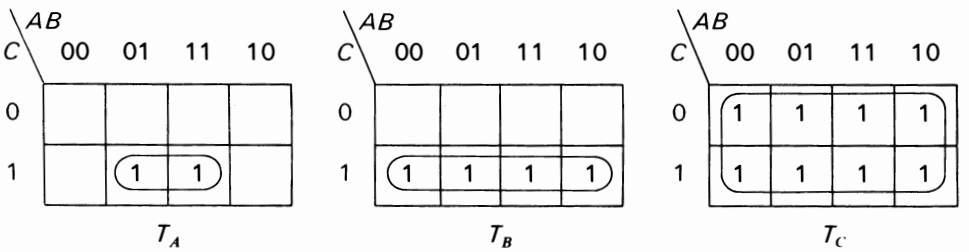


Fig. 29. *Diagramas de Karnaugh del contador binario*

La lectura de los diagramas de Karnaugh, coincidiendo con el avance da:

$$T_A = (BC)\,\text{Avc}\ ,$$

$$T_B = (C)\,\text{Avc}\ ,$$

$$T_C = (1)\,\text{Avc}\ .$$

El esquema del contador está representado en la figura 30.

Obsérvese la ley de formación de los términos de las expresiones T. Si se añade una etapa suplementaria X de peso más fuerte, tendremos:

$$T_X = (ABC)\,\text{Avc}\ ,$$

$$T_A = (BC)\,\text{Avc}\ ,$$

$$T_B = (C)\,\text{Avc}\ ,$$

$$T_C = (1)\,\text{Avc}\ .$$

Resulta pues posible generalizar para un contador de n etapas. La última etapa estará sometida al avance, la penúltima etapa estará sometida a C coincidiendo con

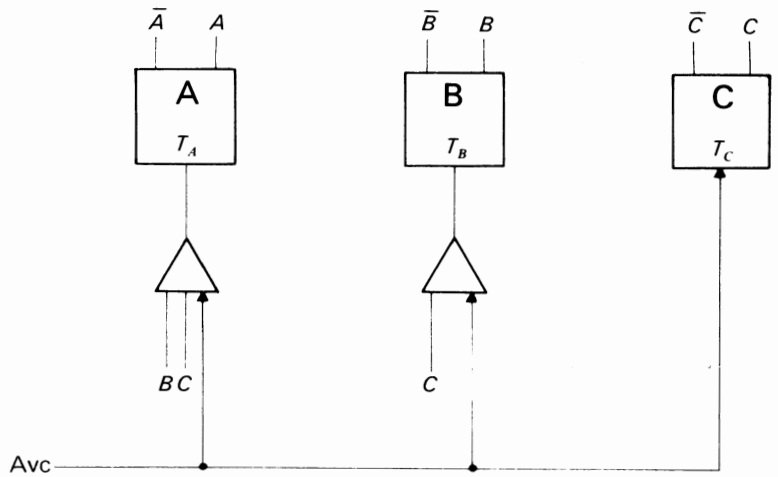


Fig. 30. *Esquema del contador binario que utiliza básculas T*

el avance, la etapa que precede estará sometida a *BC* coincidiendo con el avance, etc. Cada entrada *T* es sometida a la acción del producto lógico de las salidas verdaderas de todas las etapas situadas a su derecha coincidiendo con el avance.

3.5 Contadores que utilizan las básculas J-K

El estudio de la báscula *J-K* ha sido realizado anteriormente. Su cuadro de certeza, que fue estudiado en los capítulos 1 y 2 (Tomo I), es reproducido en la figura 31 para mayor comodidad.

Q_t	J	K	Q_{t+1}
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

Fig. 31. *Cuadro de certeza de la báscula J-K (primera versión)*

Este cuadro de certeza puede ser modificado (Fig. 32) con el fin de hacer aparecer las condiciones a aplicar a J y K para que la báscula cambie de estado o conserve su estado anterior.

Q_T	J	K	Q_{t+1}
0	0	Ø	0
0	1	Ø	1
1	0	0	1
1	Ø	1	0

Fig. 32. *Cuadro de certeza de la báscula J-K (segunda versión)*

Este es el cuadro que utilizaremos para la realización de los contadores.

3.6 Contador binario puro que utiliza las básculas J-K

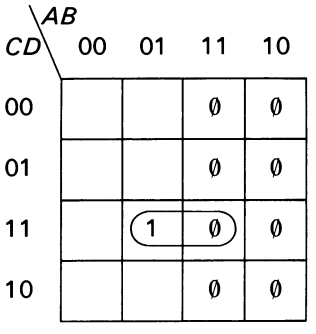
Consideremos el caso de un contador que utiliza el código binario puro para los valores decimales de 0 a 15. El contador contiene pues cuatro etapas ya que

Decimal	Avc	A	B	C	D	J_A	K_A	J_B	K_B	J_C	K_C	J_D	K_D
0	1	0	0	0	0	0	Ø	0	Ø	0	Ø	1	Ø
1	1	0	0	0	1	0	Ø	0	Ø	1	Ø	Ø	1
2	1	0	0	1	0	0	Ø	0	Ø	Ø	0	1	Ø
3	1	0	0	1	1	0	Ø	1	Ø	Ø	1	Ø	1
4	1	0	1	0	0	0	0	0	0	0	0	1	Ø
5	1	0	1	0	1	0	Ø	Ø	0	1	Ø	Ø	1
6	1	0	1	1	0	0	Ø	Ø	0	Ø	0	1	Ø
7	1	0	1	1	1	1	Ø	Ø	1	Ø	1	Ø	1
8	1	1	0	0	0	Ø	0	0	Ø	0	Ø	1	Ø
9	1	1	0	0	1	Ø	0	0	Ø	1	Ø	Ø	1
10	1	1	0	1	0	Ø	0	0	Ø	Ø	0	1	Ø
11	1	1	0	1	1	Ø	0	1	Ø	Ø	1	Ø	1
12	1	1	1	0	0	Ø	0	Ø	0	0	Ø	1	Ø
13	1	1	1	0	1	Ø	0	Ø	0	1	Ø	Ø	1
14	1	1	1	1	0	Ø	0	Ø	0	Ø	0	1	Ø
15	1	1	1	1	1	Ø	1	Ø	1	Ø	1	Ø	1
0	1	0	0	0	0								

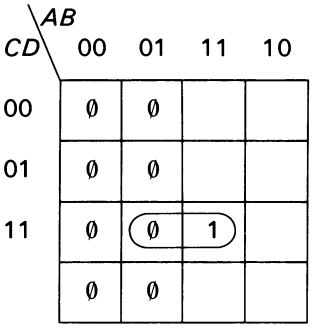
Fig. 33. *Cuadro de certeza del contador binario que utiliza las básculas J-K*

obligatoriamente se necesitan cuatro etapas para contar hasta 15. El cuadro de certeza está representado en la figura 33.

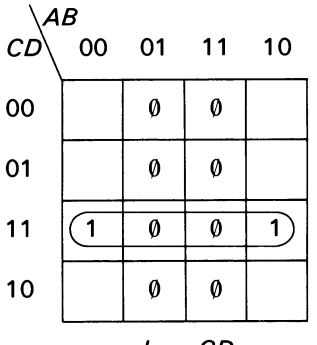
La figura 34 representa los diagramas de las diferentes entradas.



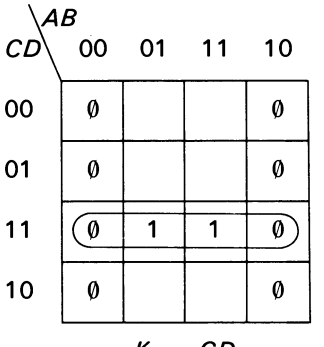
$J_A = BCD$



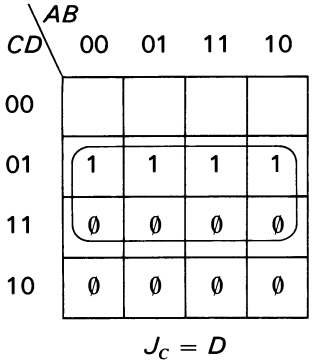
$K_A = BCD$



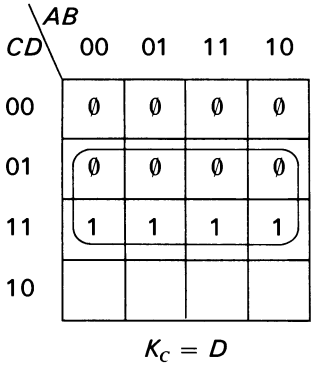
$J_B = CD$



$K_B = CD$



$J_C = D$



$K_C = D$

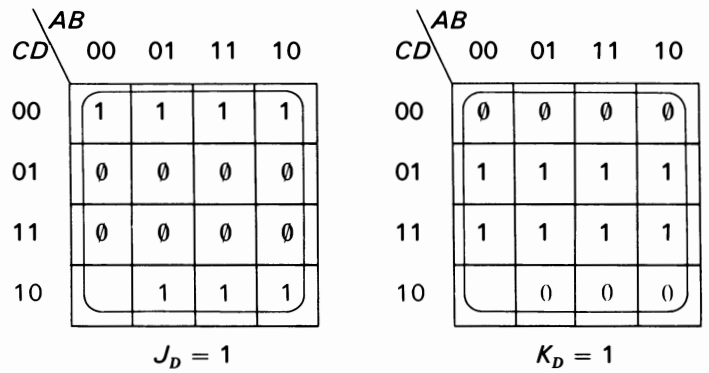


Fig. 34. *Diagrama de Karnaugh del contador binario de básculas J-K*

La lectura de los diagramas de Karnaugh da coincidiendo con el impulso de avance:

$$\begin{aligned} J_A &= K_A = (BCD)\,\text{Avc}\,, \\ J_B &= K_B = (CD)\,\text{Avc}\,, \\ J_C &= K_C = (D)\,\text{Avc}\,, \\ J_D &= K_D = (1)\,\text{Avc}\,. \end{aligned} \tag{1}$$

Refiriéndonos a lo que hemos visto anteriormente se observará que las básculas *J-K* funcionan como las básculas *T* ya que se aplican simultáneamente las mismas condiciones sobre *J* y *K*. Las básculas cambian, pues, de estado cada vez que estas condiciones son iguales a 1. El esquema de realización es el representado en la figura 35.

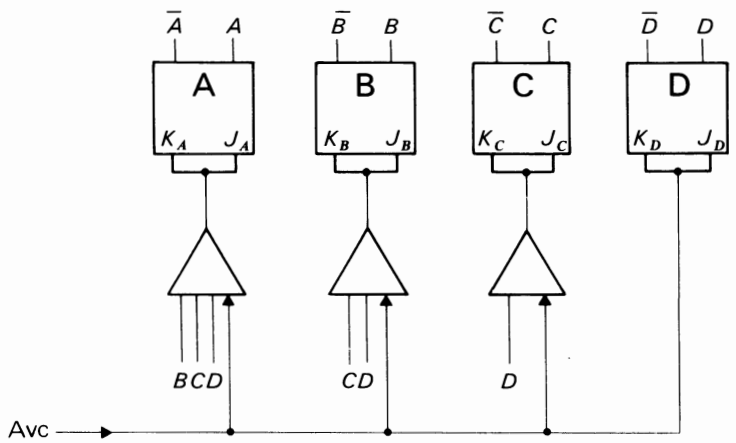


Fig. 35. *Contador binario de básculas J-K*

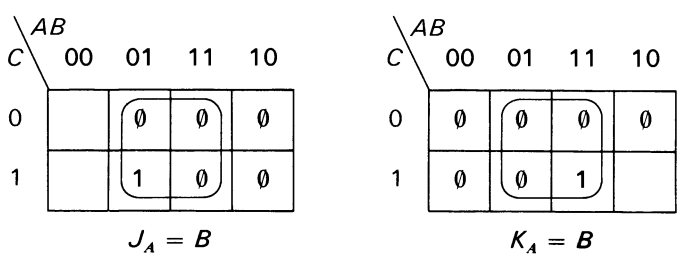
Como anteriormente se encontrará con facilidad la configuración para *n* etapas, la ley de formación de los términos de *J-K* aparece con claridad en (1).

3.6.1 *Contador binario de básculas J-K para los números impares*

En el caso del contador para la sucesión de los valores decimales 0-1-3-5-7. Los valores decimales 2, 4 y 6 no aparecen en la salida: constituyendo las *condiciones prohibidas.* En estas condiciones se puede aplicar a las entradas indiferentemente 0 ó 1, o sea Ø. El cuadro de certeza del contador está representado en la figura 36. Obsérvese la existencia de dos tipos de condiciones prohibidas (Ø), las que corresponden a los valores decimales (2, 4, 6) y las correspondientes a los valores opcionales aplicados a las entradas de las básculas para obtener la sucesión de estados deseada. Estas condiciones prohibidas son llevadas al diagrama de Karnaugh y utilizadas para la simplificación de los circuitos (fig. 37).

Decimal	Avc	*A*	*B*	*C*	J_A	K_A	J_B	K_B	J_C	K_C
0	1	0	0	0	0	Ø	0	Ø	1	Ø
1	1	0	0	1	0	Ø	1	Ø	Ø	0
3	1	0	1	1	1	Ø	Ø	1	Ø	0
5	1	1	0	1	Ø	0	1	Ø	Ø	0
7	1	1	1	1	Ø	1	Ø	1	Ø	1
0	1	0	0	0						
2	1	0	1	0	Ø	Ø	Ø	Ø	Ø	Ø
4	1	1	0	0	Ø	Ø	Ø	Ø	Ø	Ø
6	1	1	1	0	Ø	Ø	Ø	Ø	Ø	Ø

Fig. 36. *Cuadro de certeza del contador para los valores 0-1-3-5-7 que utiliza las básculas J-K*



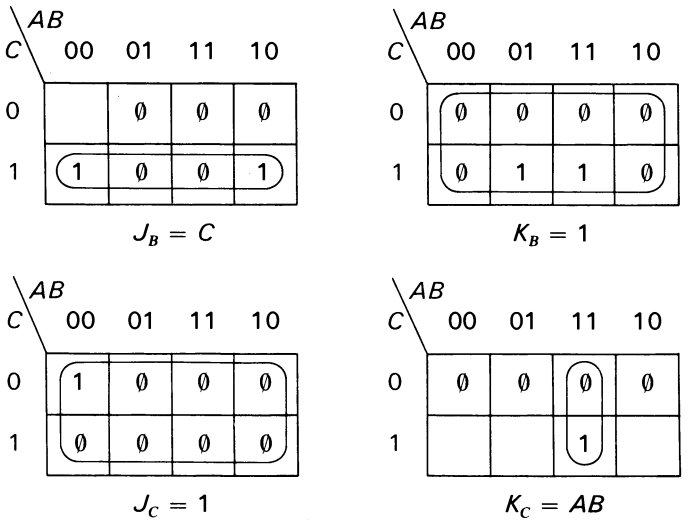


Fig. 37. *Diagramas de Karnaugh del contador de la figura 38*

La lectura de los diagramas da:

$$J_A = (B)\,\text{Avc} \quad J_B = (C)\text{Avc} \quad J_C = (1)\,\text{Avc}$$
$$K_A = (B)\,\text{Avc} \quad K_B = (1)\,\text{Avc} \quad K_C = (AB)\,\text{Avc}\ .$$

El esquema está representado en la figura 38.

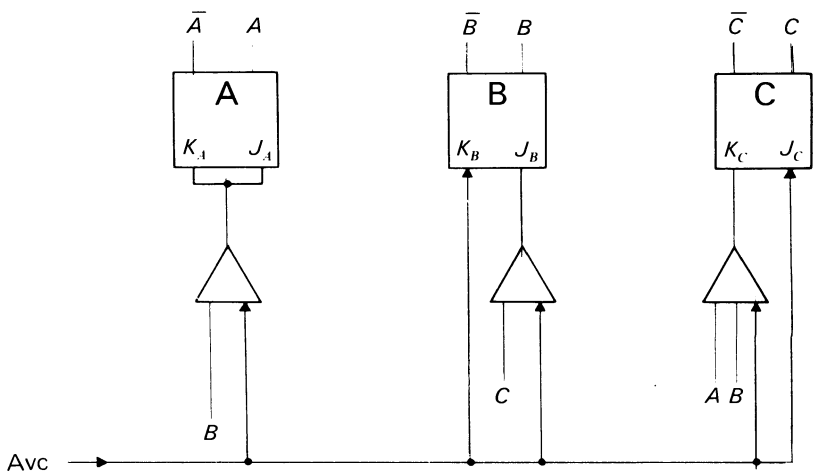


Fig. 38. *Esquema del contador de básculas J-K para los valores 0-1-3-5-7-0*

3.7 Contadores especiales

Existen muchos tipos de contadores especiales. Citemos entre otros los contadores-descontadores, los contadores cuyos estados sucesivos son función de un mando exterior, los contadores decimales y los contadores en anillos. Examinaremos los tres últimos casos.

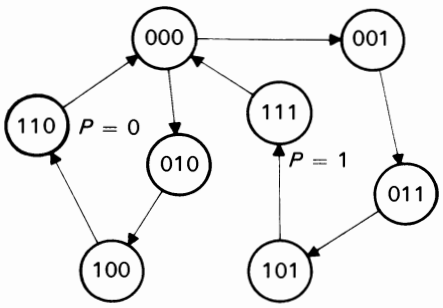


Fig. 39. *Diferentes estados del contador según el mando exterior P. Obsérvese que el mando es aplicado permanentemente sobre cada bucle.*

Decimal	Avc	A	B	C	P	T_A	T_B	T_C
0	1	0	0	0	0	0	1	0
2	1	0	1	0	0	1	1	0
4	1	1	0	0	0	0	1	0
6	1	1	1	0	0	1	1	0
0	1	0	0	0	0			
Condiciones prohibidas	1	0	0	1	0	∅	∅	∅
	1	0	1	1	0	∅	∅	∅
	1	1	0	1	0	∅	∅	∅
	1	1	1	1	0	∅	∅	∅
0	1	0	0	0	1	0	0	1
1	1	0	0	1	1	0	1	0
3	1	0	1	1	1	1	1	0
5	1	1	0	1	1	0	1	0
7	1	1	1	1	1	1	1	1
0	1	0	0	0	1			
Condiciones prohibidas	1	0	1	0	1	∅	∅	∅
	1	1	0	0	1	∅	∅	∅
	1	1	1	0	1	∅	∅	∅

Fig. 40. *Cuadro de certeza del contador a dos ciclos de la figura 39*

3.7.1 *Contador con mando exterior*

Tomemos como ejemplo un contador para la sucesión de los números pares para $P = 0$ y la sucesión de los números impares para $P = 1$. En los dos casos el contador está a cero al principio del ciclo y vuelve a cero al final del ciclo. La capacidad máxima del contador es 7.

La figura 39 representa la sucesión de los estados del contador en los dos casos $P = 0$ y $P = 1$. El contador contiene tres etapas, A, B y C. El cuadro de certeza del contador está representado en la figura 40. Se utilizan básculas T. Se imponen dos ciclos diferentes al contador según el valor de P.

Obtendremos dos series de diagramas de Karnaugh, una para $P = 0$, otra para $P = 1$ (fig. 41).

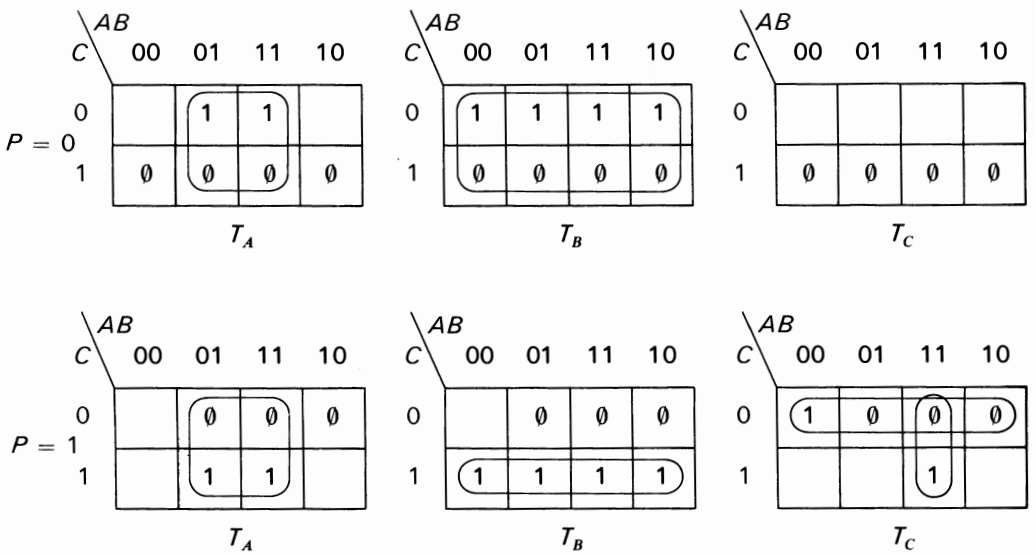


Fig. 41. *Diagramas de Karnaugh del contador de doble ciclo*

La solución será naturalmente la suma lógica de las soluciones simplificadas para $P = 0$ y $P = 1$.

$$T_A = B(P + \overline{P}).\text{Avc} = (B)\,\text{Avc}\ ,$$

$$T_B = (1.\overline{P} + CP).\text{Avc} = (\overline{P} + C)\,\text{Avc}\ ,$$

$$T_C = (\overline{C} + AB)\,P\,\text{Avc}\ .$$

El esquema es el representado en la figura 42

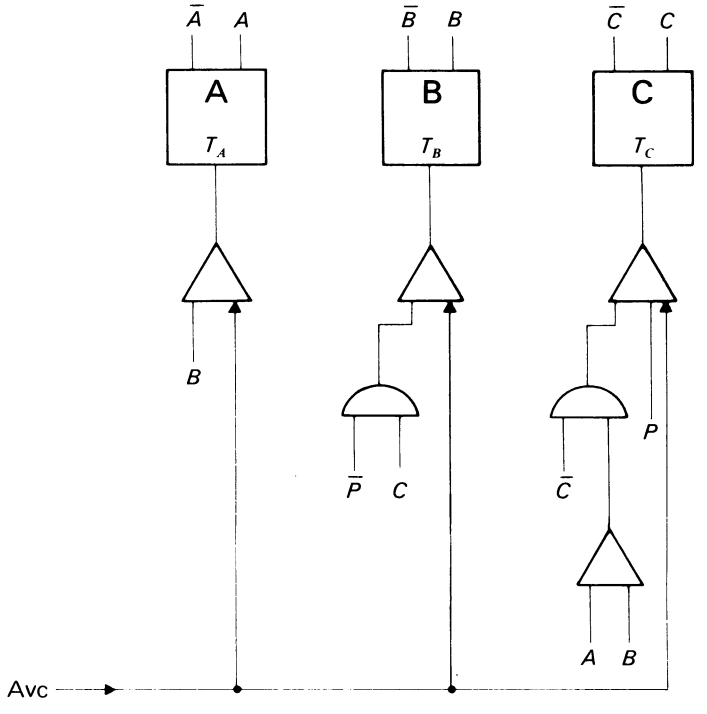


Fig. 42. *Realización del contador de doble ciclo*

Obsérvese que P o $\overline{P}$ se mantiene constantemente durante el recorrido de uno de los bucles. Por otra parte P solamente puede cambiar de estado al principio o al final del bucle.

3.7.2 *Segunda solución del problema*

Se desea ahora cambiar de bucle aplicando P o $\overline{P}$ en un momento dado. De este modo, en nuestro ejemplo se aplicará P o $\overline{P}$ solamente cuando el contador esté sobre 000. Si $P = 0$ se recorre el bucle par; si $P = 1$ se recorre el bucle impar. P se aplica únicamente cuando el contador está en 000, es decir para imponer la bifurcación. Una vez efectuado el cambio del bucle la sucesión de los estados no depende de P. El diagrama de los estados del contador está señalado en la figura 42 bis. El cuadro de certeza en la figura 43.

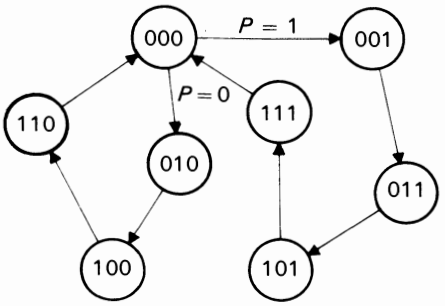


Fig. 42bis. *Diagramas de los estados del contador a doble ciclo. El mando P, solamente se aplica cuando el contador está en 000.*

Llamemos A_0, B_0, C_0, a los estados anteriores y A_1, B_1, C_1 a los nuevos estados del contador.

Decimal	Avc	A_0	B_0	C_0	P	A_1	B_1	C_1	T_A	T_B	T_C
0	1	0	0	0	0	0	1	0	0	1	0
0	1	0	0	0	1	0	0	1	0	0	1
1	1	0	0	1	0	0	1	1	0	1	0
1	1	0	0	1	1	0	1	1	0	1	0
2	1	0	1	0	0	1	0	0	1	1	0
2	1	0	1	0	1	1	0	0	1	1	0
3	1	0	1	1	0	1	0	1	1	1	0
3	1	0	1	1	1	1	0	1	1	1	0
4	1	1	0	0	0	1	1	0	0	1	0
4	1	1	0	0	1	1	1	0	0	1	0
5	1	1	0	1	0	1	1	1	0	1	0
5	1	1	0	1	1	1	1	1	0	1	0
6	1	1	1	0	0	0	0	0	1	1	0
6	1	1	1	0	1	0	0	0	1	1	0
7	1	1	1	1	0	0	0	0	1	1	1
7	1	1	1	1	1	0	0	0	1	1	1

Fig. 43. *Cuadro de certeza del contador de doble ciclo (segunda versión)*

Los diagramas de Karnaugh de las tres básculas están señalados en la figura 44. La lectura de estos diagramas da:

$$T_A = (B)\,\text{Avc}\,,$$

$$T_B = (A + B + C + \overline{P})\text{Avc}\,,$$

$$T_C = (ABC + \overline{A}\overline{B}\overline{C}P)\,\text{Avc}\,.$$

El esquema del contador es el señalado en la figura 45.

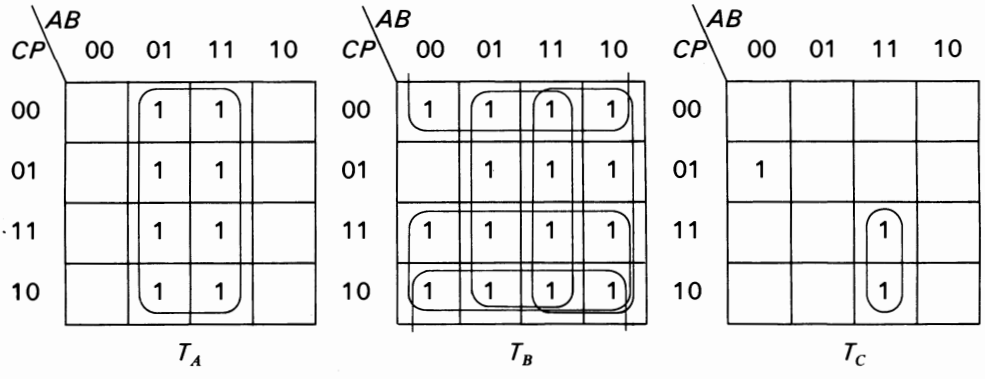


Fig. 44. *Diagramas de Karnaugh del contador de doble ciclo (segunda versión)*

3.7.3 *Generalización del caso anterior*

Examinemos ahora el caso mucho más general de las figuras 46a y 46b. En este tipo de contador el punto de partida y el punto de llegada es el estado 000. Existe una bifurcación a partir del estado 010.

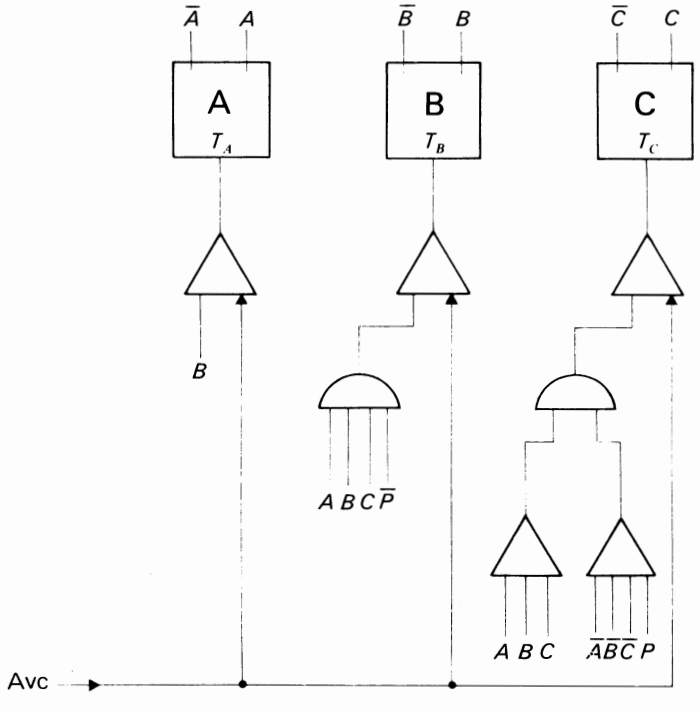


Fig. 45. *Esquema del contador de doble ciclo (segunda versión).*

Podemos proceder como anteriormente e imponer el valor $P = 0$ y $P = 1$ según el sentido y durante todo el tiempo del recorrido, o bien imponer $P = 0$ o $P = 1$ en el momento de la bifurcación, la sucesión de los estados del contador en cada bucle son independientes de P. Volviéndose a encontrar los dos casos precedentes. Obsérvese que el estado 001 no existe: es una condición prohibida.

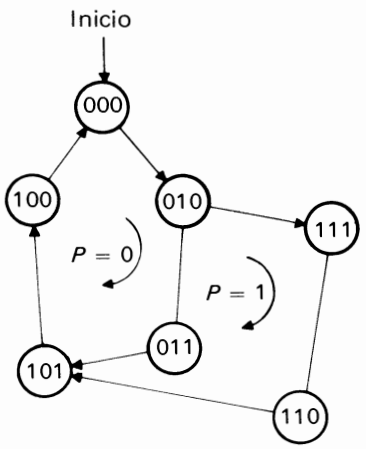


Fig. 46a.

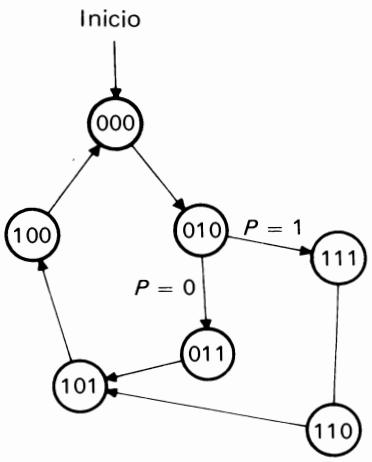


Fig. 46b.

"Diferentes modos de obtener el recorrido de los dos bucles:

a) $P = 0$ o $P = 1$ es impuesto permanentemente durante el recorrido de cada bucle.

b) $P = 0$ o $P = 1$ es impuesto solamente en el momento de la bifurcación".

5.7.3.1 *Primera versión* (Fig. 46*a*)

En el bucle correspondiente a $P = 0$ tenemos tres estados prohibidos: 001 que no existe en ninguno de los dos bucles y los estados 111 y 110 del bucle correspondiente a $P = 1$. En el bucle correspondiente a $P = 1$, hay dos estados prohibidos: 001 y 011. El cuadro de certeza del contador es el representado en la figura 47. Los diagramas de Karnaugh para $P = 0$ y $P = 1$ se dan en la figura 48. Una solución simplificada, consiste en buscar los bucles idénticos para cada par de diagramas. Eliminándose de este modo el mando P.

Decimal	Avc	*A*	*B*	*C*	*P*	T_A	T_B	T_C
0	1	0	0	0	0	0	1	0
2	1	0	1	0	0	0	0	1
3	1	0	1	1	0	1	1	0
5	1	1	0	1	0	0	0	1
4	1	1	0	0	0	1	0	0
0	1	0	0	0	0			
Estados	1	0	0	1	0	∅	∅	∅
prohibidos	1	1	1	1	0	∅	∅	∅
(1, 7, 6)	1	1	1	0	0	∅	∅	∅
0	1	0	0	0	1	0	1	0
2	1	0	1	0	1	1	0	1
7	1	1	1	1	1	0	0	1
6	1	1	1	0	1	0	1	1
5	1	1	0	1	1	0	0	1
4	1	1	0	0	1	1	0	0
0	1	0	0	0	1			
Estados prohibidos	1	0	0	1	1	∅	∅	∅
(1, 3)	1	0	1	1	1	∅	∅	∅

Fig. 47. *Cuadro de certeza del contador de la figura 46a*

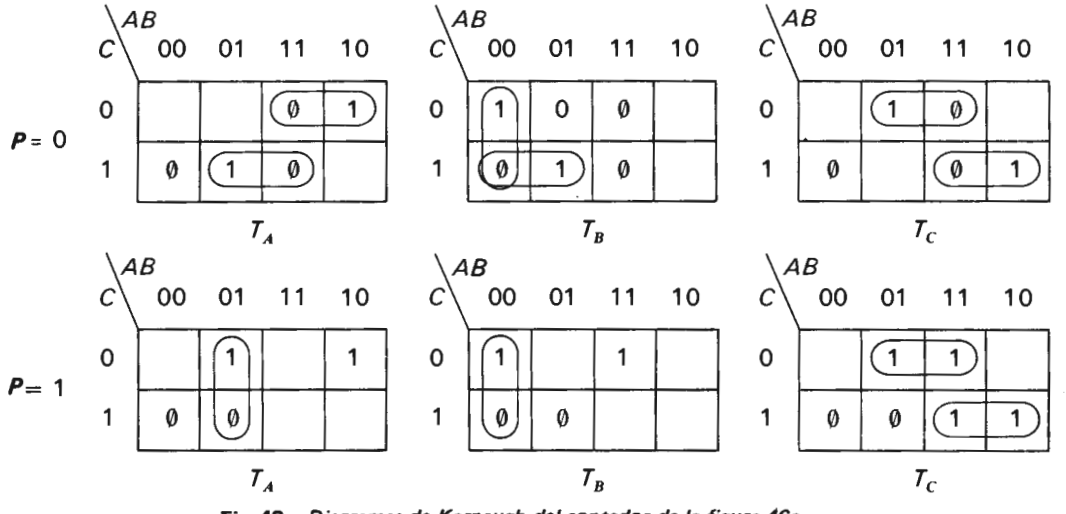


Fig. 48. *Diagramas de Karnaugh del contador de la figura 46a*

La lectura de los diagramas da:

$$T_A = [(A\overline{C} + BC)\,\overline{P} + (\overline{A}B + A\overline{B}\overline{C})\,P]\,\text{Avc}$$

después de simplificación tenemos

$$T_A = (A\overline{C}\overline{P} + BC\overline{P} + \overline{A}BP + A\overline{B}\overline{C})\,\text{Avc}$$

y para T_B y T_C

$$T_B = (\overline{A}\overline{B} + \overline{A}C\overline{P} + AB\overline{C}P)\,\text{Avc}$$
$$T_C = (B\overline{C} + AC)\,\text{Avc}\ .$$

La realización del circuito está representada en la figura 49.

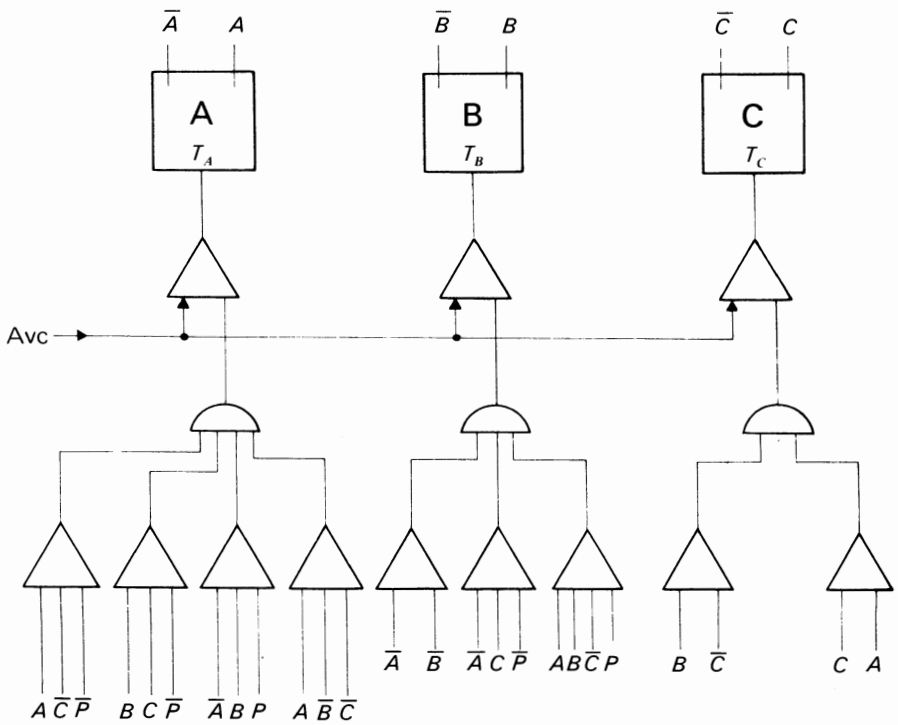


Fig. 49. *Realización del contador de la figura 46a*

3.7.3.2 *Segunda versión* (Fig. 46*b*)

En esta versión la bifurcación tiene lugar solamente en 010 según el valor de *P* *que se aplica únicamente en ese momento.* Una vez elegido el bucle, la sucesión de los estados es independiente de *P.* El cuadro de certeza del contador está representado en la figura 50. Los diagramas de Karnaugh en la figura 51.

Decimal	Avc	*A*	*B*	*C*	*P*	A_1	B_1	C_1	T_A	T_B	T_C
0	1	0	0	0	0	0	1	0	0	1	0
0	1	0	0	0	1	0	1	0	0	1	0
2	1	0	1	0	0	0	1	1	0	0	1
2	1	0	1	0	1	1	1	1	1	0	1
3	1	0	1	1	0	1	0	1	1	1	0
3	1	0	1	1	1	1	0	1	1	1	0
4	1	1	0	0	0	0	0	0	1	0	0
4	1	1	0	0	1	0	0	0	1	0	0
5	1	1	0	1	0	1	0	0	0	0	1
5	1	1	0	1	1	1	0	0	0	0	1
6	1	1	1	0	0	1	0	1	0	1	1
6	1	1	1	0	1	1	0	1	0	1	1
7	1	1	1	1	0	1	1	0	0	0	1
7	1	1	1	1	1	1	1	0	0	0	1
Estados prohibidos 1	1	0	0	1	0				∅	∅	∅
1	1	0	0	1	1				∅	∅	∅

Fig. 50. *Cuadro de certeza del contador de la figura 46b*

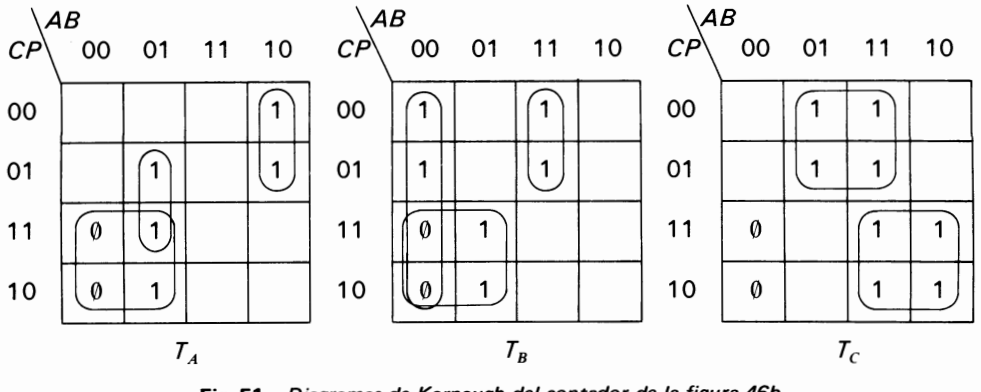


Fig. 51. *Diagramas de Karnaugh del contador de la figura 46b*

La lectura de estos diagramas da, coincidiendo con el avance:

$$T_A = (\overline{A}C + \overline{A}BP + A\overline{B}\overline{C})\text{Avc}\,,$$
$$T_B = (\overline{A}\overline{B} + \overline{A}C + AB\overline{C})\text{Avc}\,,$$
$$T_C = (AB + AC + B\overline{C})\text{Avc}\,.$$

El esquema de realización está representado en la figura 52. A, B, C son los estados anteriores, A_1, B_1. C_1 los nuevos estados del contador con el impulso de avance siguiente.

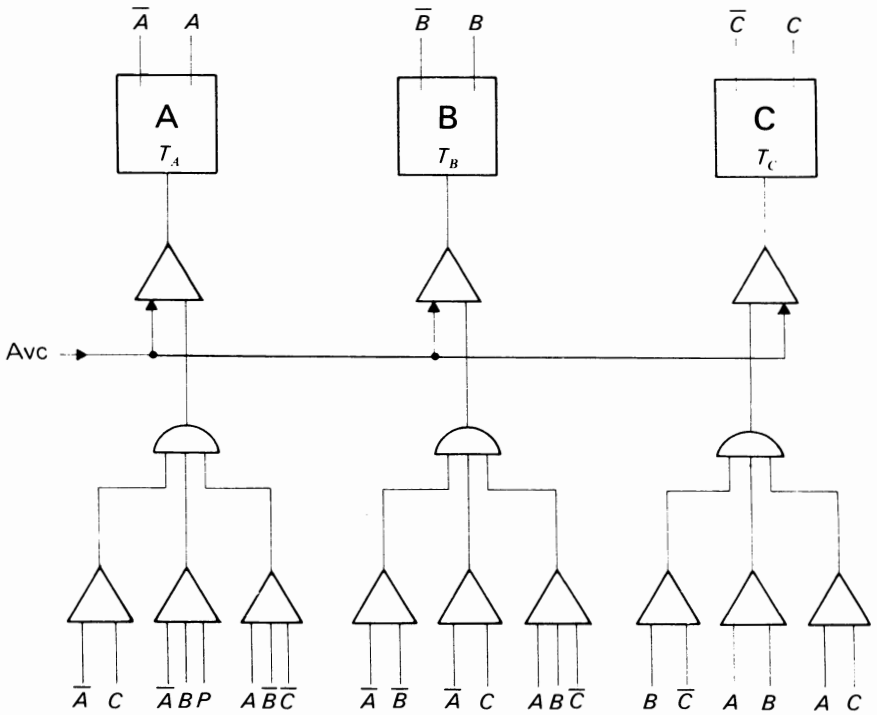


Fig. 52. *Realización del contador de la figura 46b.*

3.8 Contadores decimales

Un contador decimal (o de base 10) es un contador que da los 10 símbolos de 0 a 9 en el código 1-2-4-8 (fig. 53). Este contador comprende cuatro básculas. Puede contar de 0 a 15 incluido. Los seis estados de 10 a 15 son prohibidos.

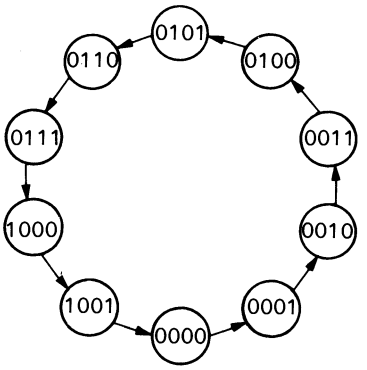


Fig. 53. *Estados sucesivos de un contador decimal*

Decimal	Avc	*A*	*B*	*C*	*D*	T_A	T_B	T_C	T_D
0	1	0	0	0	0	0	0	0	1
1	1	0	0	0	1	0	0	1	1
2	1	0	0	1	0	0	0	0	1
3	1	0	0	1	1	0	1	1	1
4	1	0	1	0	0	0	0	0	1
5	1	0	1	0	1	0	0	1	1
6	1	0	1	0	1	0	0	0	1
7	1	0	1	1	1	1	1	1	1
8	1	1	0	0	0	0	0	0	1
9	1	1	0	0	1	1	0	0	1
	0	1	0	0	0				
Condiciones prohibidas	1	1	0	1	0	∅	∅	∅	∅
	1	1	0	1	1	∅	∅	∅	∅
	1	1	1	0	0	∅	∅	∅	∅
	1	1	1	0	1	∅	∅	∅	∅
	1	1	1	1	0	∅	∅	∅	∅
	1	1	1	1	1	∅	∅	∅	∅

Fig. 54. *Cuadro de certeza del contador decimal (Fig. 53)*

Las básculas utilizadas son del tipo T. El cuadro de certeza está representado en la figura 54. Los diagramas de Karnaugh en la figura 55. La lectura de estos

diagramas da, coincidiendo con el avance:

$$T_A = D(A + BC)\text{Avc}\,,$$
$$T_B = (CD)\text{Avc}\,,$$
$$T_C = (\bar{A}D)\text{Avc}\,,$$
$$T_D = (1)\text{Avc}\,.$$

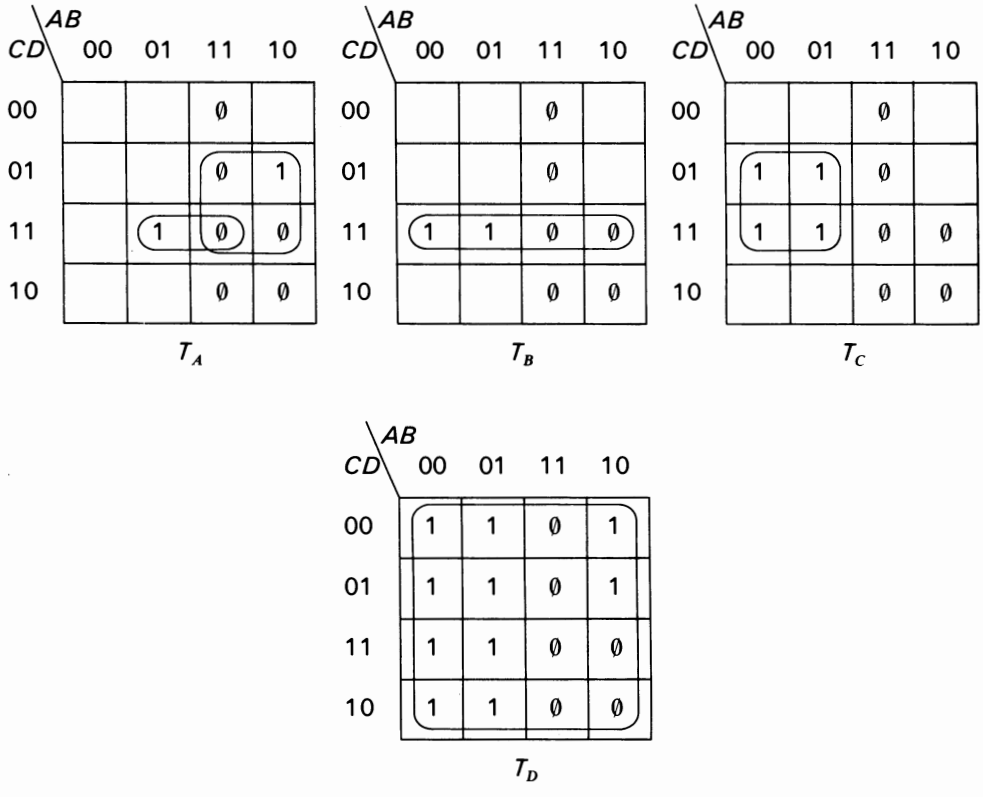


Fig. 55. *Diagramas de Karnaugh del contador decimal de la figura 53*

El esquema de realización está representado en la figura 56.

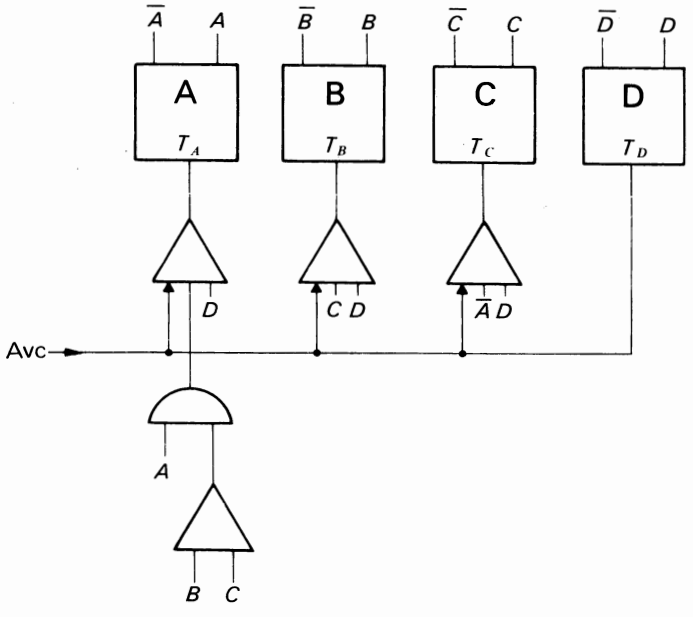


Fig. 56. *Realización del contador decimal de la figura 53*

3.9 Contadores en anillo y registros de desplazamientos

Se entiende por contador en anillo un contador que contiene tantas básculas como símbolos hay en la base de numeración elegida, estando cada báscula en 1 para uno de los símbolos del sistema. Como ejemplo de aplicación consideramos un contador de base 10, es decir que comprende diez básculas designadas por A_0, A_1, A_2, ... A_8, A_9.

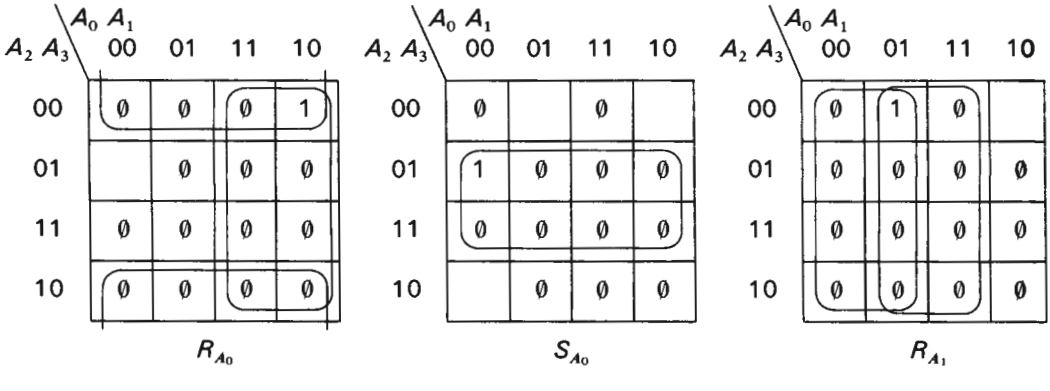
Decimal	Avc	A_0	A_1	A_2	A_3		A_9
0	1	1	0	0	0		0
1	1	0	1	0	0		0
2	1	0	0	1	0		0
3	1	0	0	0	1		0
	:	:	:	:	:		:
	.	.	.	.	.		.
9	1	0	0	0	0		1
0	1	1	0	0	0		0

Fig. 57. *Cuadro de certeza de un contador en anillo*

Por definición diremos que el contador está en cero cuando la báscula A_0 está en 1. El contador estará en 1 cuando la báscula A_1 está en 1, y así sucesivamente. Esto equivale a desplazar sucesivamente un 1 sobre cada báscula a cada impulso de avance. El cuadro de certeza de este tipo de contador está representado en la figura 57.

Decimal	Avc	A_0	A_1	A_2	A_3	R_{A_0}	S_{A_0}	R_{A_1}	S_{A_1}	R_{A_2}	S_{A_2}	R_{A_3}	S_{A_3}
0	1	1	0	0	0	1	0	0	1	∅	0	∅	0
1	1	0	1	0	0	∅	0	1	0	0	1	∅	0
2	1	0	0	1	0	∅	0	∅	0	1	0	0	1
3	1	0	0	0	1	0	1	∅	0	∅	0	1	0
0	1	1	0	0	0								
Condiciones imposibles	1	0	0	0	0	∅	∅	∅	∅	∅	∅	∅	∅
	1	0	0	1	1	∅	∅	∅	∅	∅	∅	∅	∅
	1	0	1	0	1	∅	∅	∅	∅	∅	∅	∅	∅
	1	0	1	1	0	∅	∅	∅	∅	∅	∅	∅	∅
	1	0	1	1	1	∅	∅	∅	∅	∅	∅	∅	∅
	1	1	0	0	1	∅	∅	∅	∅	∅	∅	∅	∅
	1	1	0	1	0	∅	∅	∅	∅	∅	∅	∅	∅
	1	1	0	1	1	∅	∅	∅	∅	∅	∅	∅	∅
	1	1	1	0	0	∅	∅	∅	∅	∅	∅	∅	∅
	1	1	1	0	1	∅	∅	∅	∅	∅	∅	∅	∅
	1	1	1	1	0	∅	∅	∅	∅	∅	∅	∅	∅
	1	1	1	1	1	∅	∅	∅	∅	∅	∅	∅	∅

Fig. 58. *Cuadro de certeza de un contador en anillo de cuatro etapas*



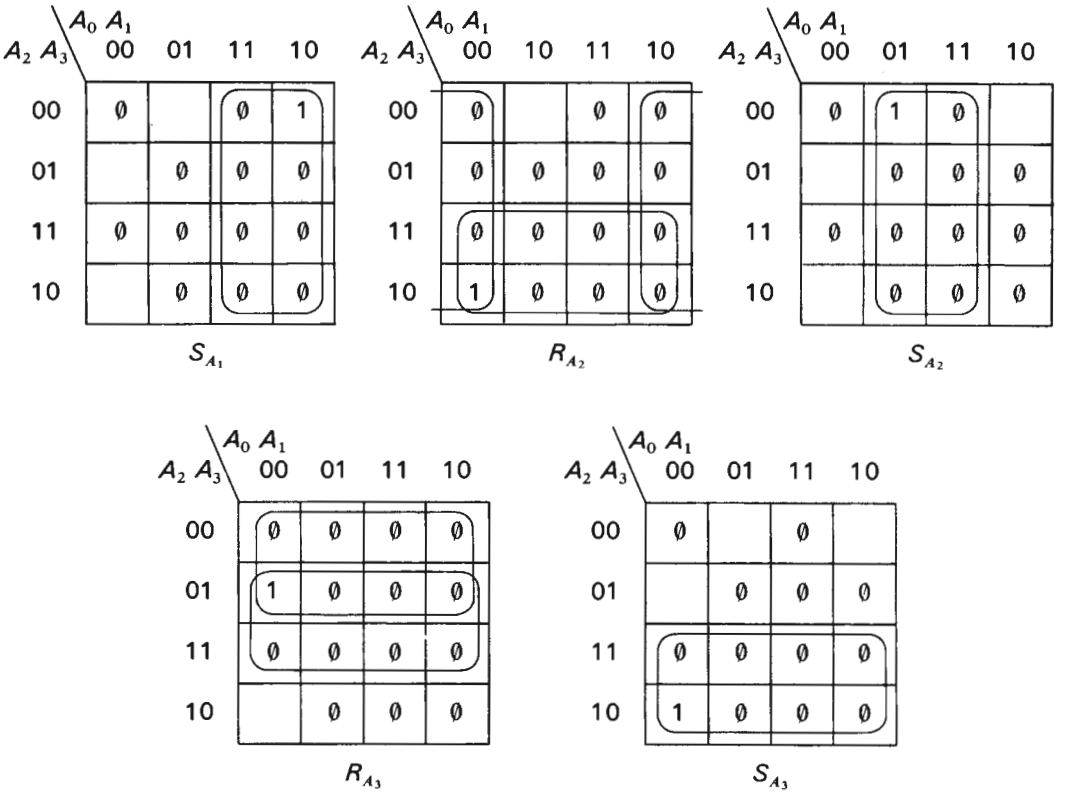


Fig. 59. *Diagramas de Karnaugh del contador en anillo*

Se utilizan básculas *R-S* para realizar fácilmente la puesta a cero del contador antes de la introducción de un 1 en la báscula A_0. "*Es pues necesario inicializar el contador*". Examinemos el comportamiento de un contador en anillo de cuatro básculas: el resultado se aplicará a un contador de N etapas. El cuadro de certeza está representado en la figura 58. Los diagramas de Karnaugh en la figura 59.

La lectura de estos diagramas da:

$$R_{A_0} = A_0 \text{ ó } \overline{A}_3 \qquad R_{A_2} = A_2 \text{ ó } \overline{A}_1$$
$$S_{A_0} = A_3 \qquad S_{A_2} = A_1$$
$$R_{A_1} = A_1 \text{ ó } \overline{A}_0 \qquad R_{A_3} = A_3 \text{ ó } \overline{A}_2$$
$$S_{A_1} = A_0 \qquad S_{A_3} = A_2 .$$

Se observa que existen dos valores para R_{A_1}. Escogeremos por razones de simetría con S_{A_1}, el valor barrado. En estas condiciones obtenemos

$$R_{A_i} = \overline{A_{i-1}},$$
$$S_{A_i} = A_{i-1},$$

De donde se obtiene:

$$R_{A_i} = \overline{S_{A_i}},$$
$$S_{A_i} = \overline{R_{A_i}}.$$

El esquema del contador en anillo está representado en la figura 60. Obsérvese que en este circuito todas las etapas están unidas entre ellas del mismo modo: la salida "verdad" de la etapa precedente está unida a la entrada "verdad" de la etapa siguiente. La salida complementaria de la etapa precedente está unida a la entrada complementaria de la etapa siguiente. Finalmente la última y la primera etapas están unidas entre ellas del mismo modo: razón por la cual este circuito es llamado contador *en "anillo"*.

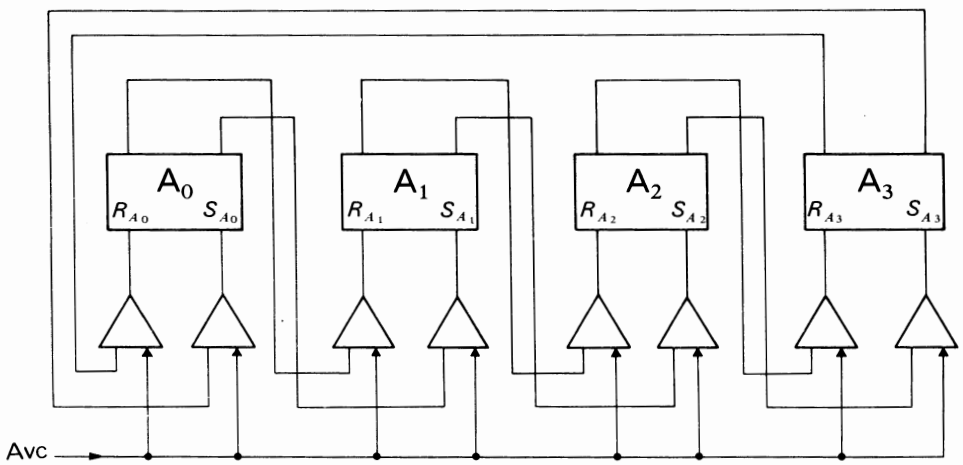


Fig. 60. *Contador en anillo de cuatro etapas*

3.9.1 *Principales propiedades de los contadores en anillo*

– Posibilidades de realizar contadores decimales de lectura directa.

– La información contenida en el contador se desplaza de una posición a cada impulso de avance y es restituida a la entrada. *La información no se pierde sino que circula permanentemente en el contador.* Es lo que se llama una "memoria circulante".

– Separando las bornas de salida se tiene un *"registro de desplazamientos"*: la información se desplaza de una posición a la derecha o a la izquierda, según las conexiones entre las etapas, a cada impulso de avance. Cuando se han producido un número de impulsos de avance igual al número de etapas, el contenido del registro ha desaparecido.

Un contador en anillo o un registro de desplazamientos, obligatoriamente debe ser inicializado, es decir que debe contener una información antes de la aplicación del primer impulso de avance. Es pues necesario prever circuitos anexos que permitan la introducción del dato antes de comenzar el contado. Si el contador no tiene información siempre quedará a cero cualquiera que sea el número de impulsos de avance aplicados.

3.9.2 *Registros de desplazamientos*

Tal como acabamos de ver un registro de desplazamientos, desplaza su contenido de una posición a la derecha o a la izquierda a cada impulso de avance. Solamente se diferencia del contador en anillo por la configuración de la primera y la última etapa. Existen los registros de desplazamientos a la derecha y a la izquierda.

1. *Registro de desplazamientos a la derecha sin conservación del primer dígito*

En este caso el contenido del contador se desplaza de una posición a la derecha introduciendo un cero en la primera báscula. El esquema de realización está representado en la figura 61.

Se observará que el paso de un dígito de orden i a la etapa de orden $i + 1$ es provocada por el estado de la báscula precedente y se efectúa al paso del flanco negativo del impulso de avance.

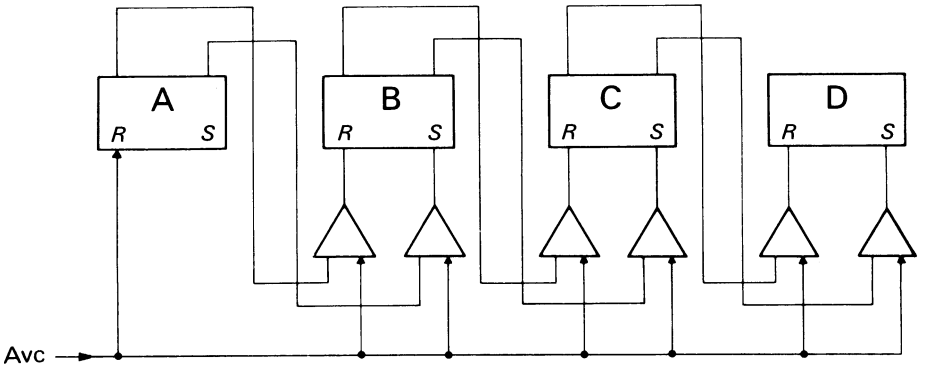


Fig. 61. *Registro de desplazamientos a la derecha sin conservación del primer dígito*

2. *Registro de desplazamientos a la derecha con conservación del primer dígito.*

En ciertos casos es necesario desplazar un número binario de varias posiciones a la derecha conservando el primer dígito. Este es el caso del desplazamiento de los números negativos representados por sus complementos a dos. En este caso el

contenido de la primera báscula debe ser preservado a cada impulso de avance. El esquema de realización es el representado en la figura 62.

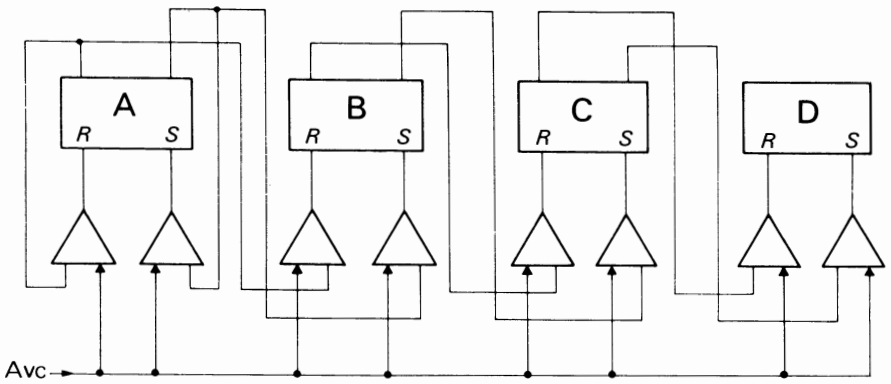


Fig. 62. *Registro de desplazamientos a la derecha con conservación del primer dígito*

3. *Registro de desplazamientos a la izquierda*

En el caso de un desplazamiento a la izquierda, el contenido de un registro es desplazado de una posición a la izquierda a cada impulso de avance. Se introduce un cero sistemáticamente en el último peso. El esquema está representado en la figura 63.

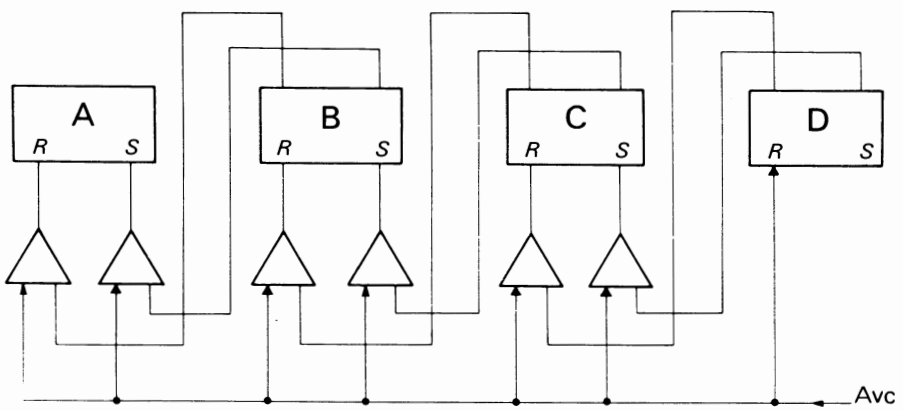


Fig. 63. *Registro de desplazamientos a la izquierda*

NOTA. Un mismo registro puede realizar desplazamientos a la izquierda o a la derecha. Basta con prever las conexiones necesarias así como dos mandos

suplementarios, operación desplazamiento a la derecha (Op Dpz dr) y operación desplazamiento a la izquierda (Op Dpz id) que controlan el sentido del desplazamiento. Más adelante estudiaremos un registro que combina las dos funciones en *"lógica dinámica"*.

3.9.3 *Representación de contadores, registros de desplazamientos, contadores en anillo.*

Ha sido desarrollada una simbología para la representación de un registro, de un contador, de un registro de desplazamientos, de un contador en anillo. Todos estos elementos son representados del mismo modo: un rectángulo dividido en tantos casilleros binarios como pesos tenga el contador. Caso de un registro que toma en memoria un número binario. Si se trata de un contador o de un registro de desplazamientos se aplica un avance a la derecha o a la izquierda según el sentido del desplazamiento (Fig. 64). Si se trata de un contador en anillo la salida está bucleada sobre la entrada.

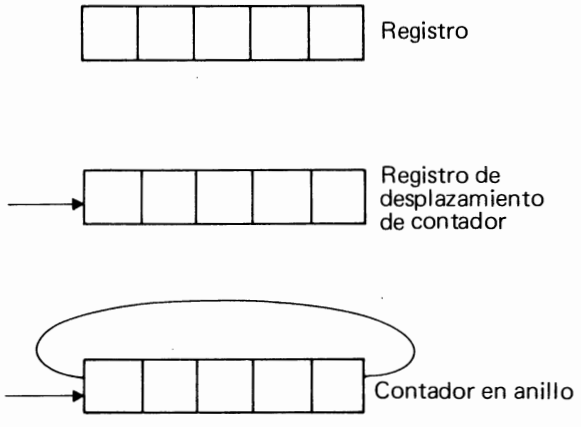


Fig. 64. *Símbolos de un registro, de un contador y de un contador en anillo*

3.9.4 *Propiedades de los registros de desplazamientos*

Desplazando un número binario de una posición a la derecha se divide su equivalente decimal por dos. Inversamente desplazando un número binario de una posición a la izquierda se multiplica su equivalente decimal por dos. Esto solamente es válido en el código binario puro.

3.9.5 *Memoria circulante con inversión de datos*

Cruzando los hilos de salida de la última etapa de un contador en anillo, es decir uniendo la salida "verdad" de la última etapa a la entrada R de la primera y la salida complementaria de la última etapa a la entrada S de la primera etapa, sin cambiar nada en las otras conexiones entre las etapas, se obtiene un contador en

anillo con inversión de los datos. Si el contador tiene cuatro pesos, al cabo de cuatro impulsos de avance se encuentra la información complementada a 1 en el contador. Después de cuatro nuevos impulsos de avance la información inicial se vuelve a encontrar en el contador: se ha realizado una memoria circulante con inversión de la información (complemento a 1) cada cuatro impulsos de avance. En este caso el contador utiliza el código "creciente". El contador se rellena de 1 a cada impulso de avance, después se vacía. El esquema de este contador está representado en la figura 66 en el caso de una inicialización.

Decimal	Avc	A_0	A_1	A_2	A_3		A_9
0	1	1	0	0	0		0
1	1	1	1	0	0		0
2	1	1	1	1	0		0
3	1	1	1	1	1		0
⋮	⋮	⋮	⋮	⋮	⋮		⋮
9	1	1	1	1	1		1

Fig. 65. *Cuadro de certeza del código creciente*

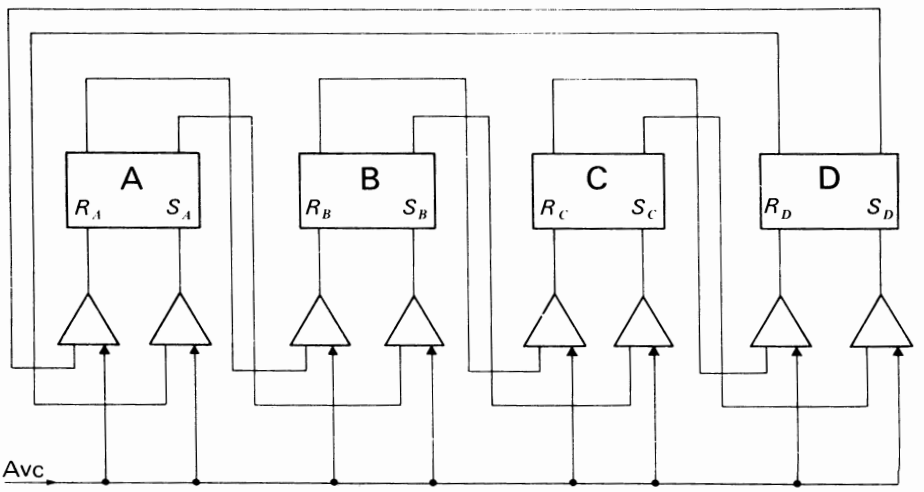


Fig. 66. *Contador en anillo que utiliza el código creciente*

Con este código es posible dividir por dos el número de etapas de un contador. Así, para contar hasta el valor decimal 10, basta con rellenar el contador de 1

hasta 5, después vaciar el contador hasta 10. Este código está representado en la figura 67.

Decimal	Avc	A_1	A_2	A_3	A_4	A_5
0	1	0	0	0	0	0
1	1	1	0	0	0	0
2	1	1	1	0	0	0
3	1	1	1	1	0	0
4	1	1	1	1	1	0
5	1	1	1	1	1	1
6	1	0	1	1	1	1
7	1	0	0	1	1	1
8	1	0	0	0	1	1
9	1	0	0	0	0	1
0	1	0	0	0	0	0

Fig. 67. *Código creciente utilizado en un contador decimal de cinco básculas*

3.10 Contadores de funcionamiento dinámico

El funcionamiento de los circuitos estudiados hasta ahora es un funcionamiento estático. Cada báscula impone a cada circuito "Y" un nivel que prepara el paso al estado siguiente, teniendo lugar éste al cesar el impulso de avance. En los circuitos anteriores es posible suprimir todos los circuitos "Y" así como las conexiones salidas complementarias-entradas *R*. Obteniendo de este modo un contador simplificado. Las básculas cambian de estado con uno de los frentes de impulsos. Supongamos que cambian con el frente negativo de los impulsos. El cambio de estado de la báscula precedente es el que impone el estado de la báscula siguiente. La figura 68 representa el esquema de un contador de cuatro etapas realizado de este modo con básculas *J-K*.

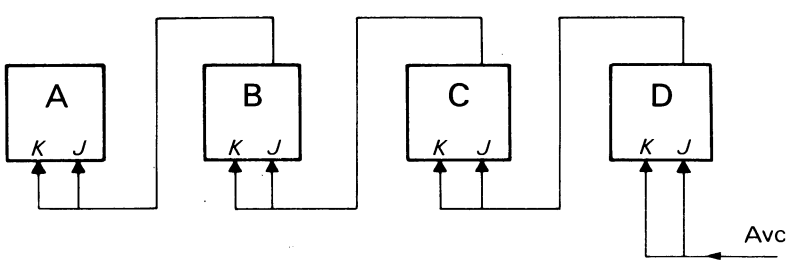


Fig. 68. *Contador binario dinámico. El cambio de las básculas B, C y D es impuesto por la báscula precedente*

En la figura 69 se representa el cuadro de certeza del contador. Reflejando los puntos siguientes:

– La última etapa *D* cambia de estado a cada impulso de avance.
– Las etapas *B*, *C* y *D* cambian de estado cada vez que las etapas precedentes pasan de 1 a 0.

Este contador puede igualmente realizarse con básculas *T*, el esquema es representado en la figura 70.

Decimal	Avc	*A*	*B*	*C*	*D*
0	1	0	0	0	0
1	1	0	0	0	1
2	1	0	0	1	0
3	1	0	0	1	1
4	1	0	1	0	0
5	1	0	1	0	1
6	1	0	1	1	0
7	1	0	1	1	1
8	1	1	0	0	0
9	1	1	0	0	1
10	1	1	0	1	0
11	1	1	0	1	1
12	1	1	1	0	0
13	1	1	1	0	1
14	1	1	1	1	0
15	1	1	1	1	1

Fig. 69. *Cuadro de certeza de un contador binario dinámico*

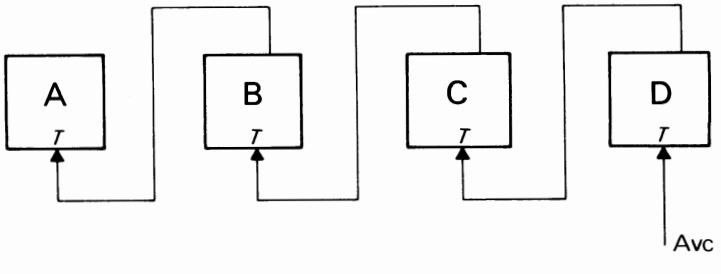


Fig. 70. *Contador binario dinámico realizado con básculas T.*

El contenido del contador aumenta de una unidad por cada impulso de avance que se aplica sobre la báscula *D*. El paso de una báscula del estado 1 al estado 0, provoca el cambio de estado de la báscula siguiente. Uniendo la salida complementaria de cada báscula a la entrada *T* de la báscula siguiente, se realiza un contador cuyo contenido disminuye en una unidad a cada impulso de avance: es lo que se llama descontador (Fig. 71).

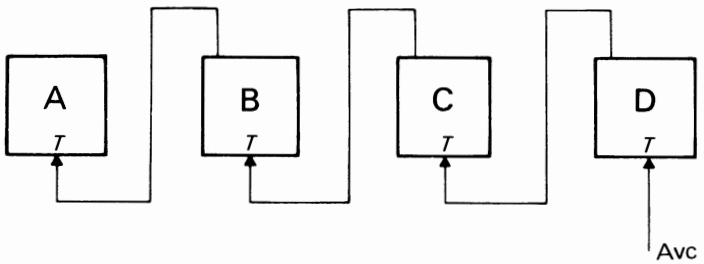


Fig. 71. *Descontador binario realizado con báscula T.*

3.11 Registros de desplazamientos dinámicos

Es igualmente posible realizar registros de desplazamientos con lógica dinámica. En este caso, como anteriormente, es el paso de 1 a 0 de la báscula precedente el que efectúa el desplazamiento del dato.

3.11.1 *Desplazamientos a la derecha con conservación del primer dígito*

Se procede del modo siguiente. El impulso de avance pone a cero el conjunto del registro. Las básculas que están en 1 pasan a 0 forzando al mismo tiempo el paso de la báscula siguiente a 1. De este modo se desplaza el contenido de un registro de una posición a la derecha. Las básculas son del tipo *R-S* y la salida "verdad" de una está unida a la entrada "verdad" de la otra. La salida "verdad" de la primera báscula está conectada a su entrada "verdad", lo que permite la conservación del primer dígito. El esquema está representado en la figura 72.

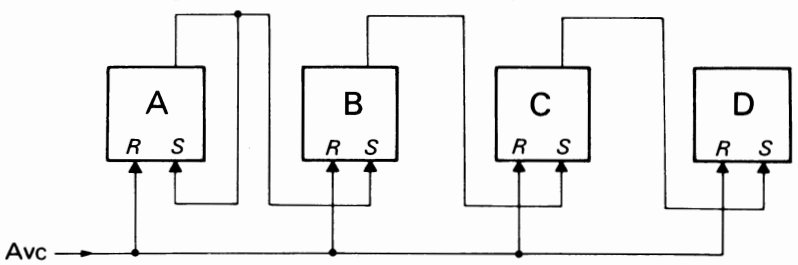


Fig. 72. *Registro de desplazamientos a la derecha que conserva el primer dígito en lógica dinámica*

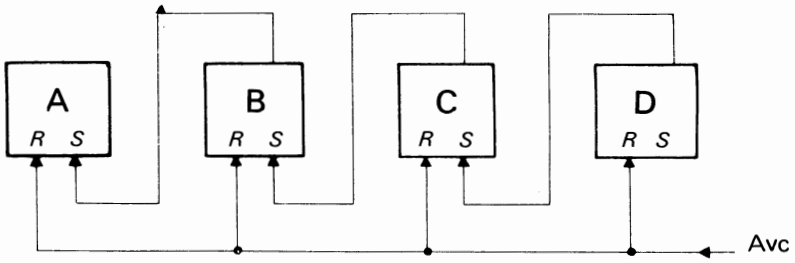


Fig. 73. *Registro de desplazamientos a la izquierda que no conserva el último dígito en lógica dinámica.*

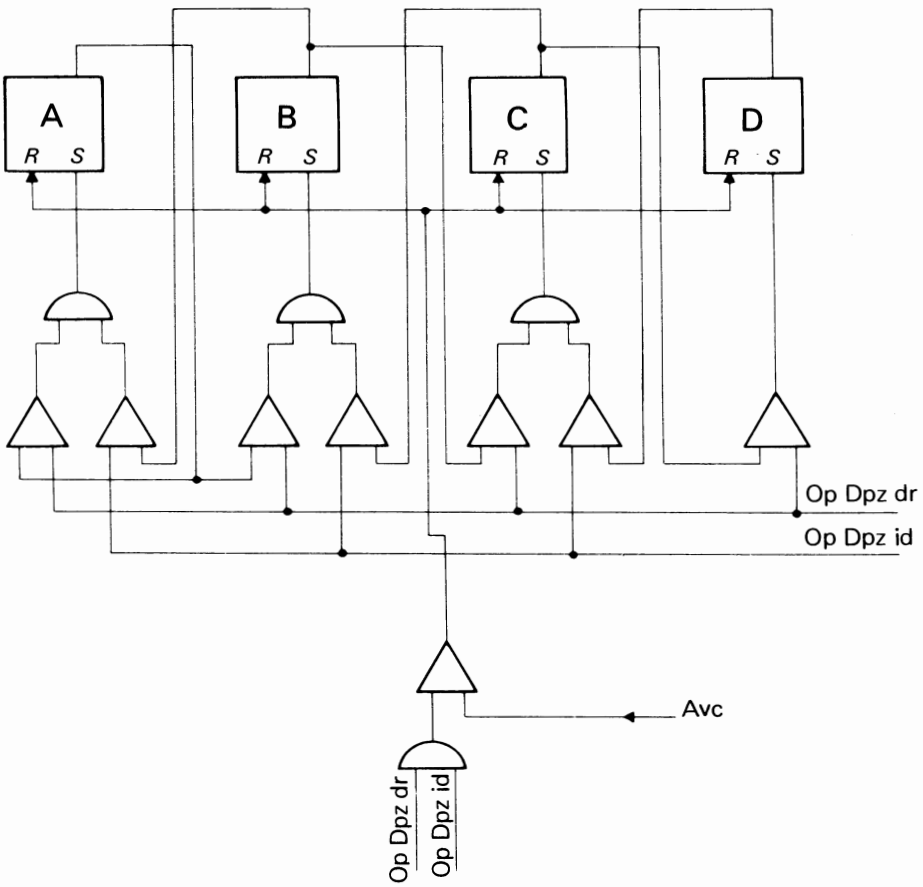


Fig. 74. *Registro de desplazamientos a derecha e izquierda en lógica dinámica*

3.11.2 *Desplazamientos a la izquierda*

Para hacer desplazamientos a la izquierda, se procede del mismo modo. Las conexiones entre las básculas se efectúan ahora de derecha a izquierda. El modo de funcionamiento es idéntico al anterior. No obstante en un desplazamiento a la izquierda no se conserva el último peso del registro (2^0). Este es sistemáticamente reemplazado por cero a cada impulso de avance. El esquema está representado en la figura 73.

3.11.3 *Desplazamientos a la derecha y a la izquierda*

Pueden ser reunidas en un solo registro las dos funciones de desplazamientos a la derecha y a la izquierda. Es necesario, en este caso, prever dos mandos suplementarios para efectuar la selección. Se trata de los mandos Op Dpz dr y Op Dpz id. El esquema está representado en la figura 74.

3.12 Contador decimal biquinario

Este dispositivo utiliza un contador en anillo del primer tipo de desplazamiento de "1" con puesta a cero de la etapa precedente. A este contador se añade una báscula *T* que cambia de estado a cada cinco impulsos de avance. Durante los impulsos de avance de 0 a 4 la báscula *T* permanece en cero, de 5 a 10 está en 1. En los dos casos el contador cuenta según el mismo código. La última báscula permite determinar si el número representado está comprendido entre cero y 4 ó entre 5 y 10. Este código se llama también 0-1-2-3-4-5. Es un código ponderado. La figura 75 representa el cuadro de certeza del contador. El circuito (Fig. 76) comprende seis básculas: cinco básculas *R-S* para el contador en anillo, una báscula *T* para la última etapa. Esta cambiará de estado cada vez que la báscula *R-S* anterior pase de 1 a 0.

Decimal	Avc	A_0	A_1	A_2	A_3	A_4	A_5
0	1	1	0	0	0	0	0
1	1	0	1	0	0	0	0
2	1	0	0	1	0	0	0
3	1	0	0	0	1	0	0
4	1	0	0	0	0	1	0
5	1	1	0	0	0	0	1
6	1	0	1	0	0	0	1
7	1	0	0	1	0	0	1
8	1	0	0	0	1	0	1
9	1	0	0	0	0	1	1
0	1	1	0	0	0	0	0

Fig. 75. *Cuadro de certeza (código utilizado) del contador decimal biquinario*

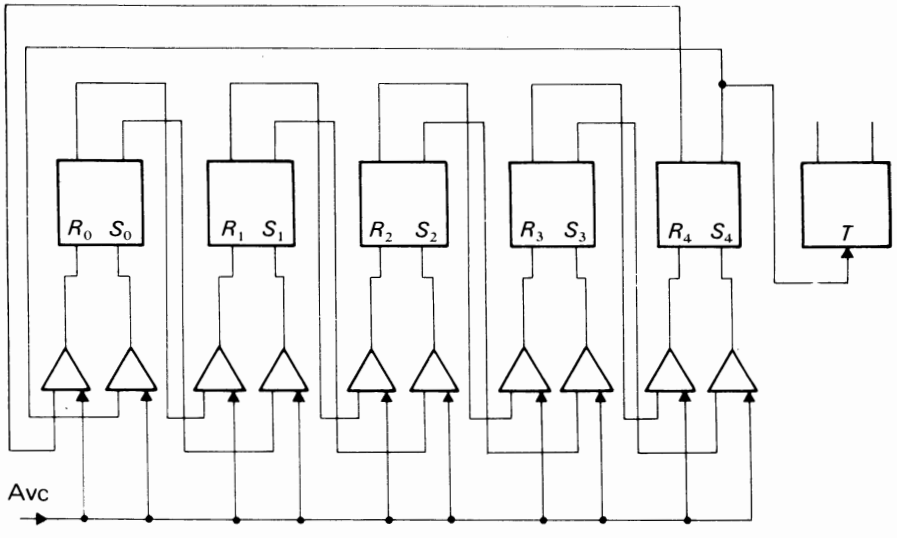


Fig. 76. *Realización de un contador decimal biquinario*

4. Transferencia de las informaciones en un calculador

Un calculador es un sistema de tratamiento de datos. Estos datos, habitualmente son introducidos en la memoria del calculador antes del cálculo. Seguidamente el calculador procesa estos datos en memoria haciéndoles seguir cierto número de transformaciones: transferencia de la memoria hacia un registro de la máquina, transferencia de un registro a otro, adición, sustracción, división, operaciones lógicas, etc.

En todos los casos es necesario efectuar una transferencia de los datos entre dos o más registros de la máquina.

Estas transferencias pueden estar acompañadas de operaciones aritméticas o lógicas. Se puede realizar, en efecto, una operación lógica o aritmética, al mismo tiempo que se efectúa una transferencia. Volveremos sobre este punto al estudiar la unidad aritmética y lógica. Por el momento examinaremos las transferencias de datos entre dos registros en *"paralelo" o en "serie"* con posibilidad de realizar *operaciones lógicas.*

4.1 Simbolismo utilizado

La transferencia de un dato generalmente tiene lugar bajo una orden del calculador. Llamemos *P* a esta orden. Los registros que participan en la transferencia son designados por letras mayúsculas: *A, B, C...* El contenido de un

registro es designado por una letra mayúscula entre paréntesis: *(A)* designa el contenido del registro *A*. (A_i) designa el i-ésimo bit del registro *A*. La operación de transferencia del contenido del registro *A* a un registro *B* se escribirá del modo siguiente:

$$P : (A) \Rightarrow B .$$

La letra *P* seguida de : indica que la transferencia se efectúa bajo la aparición del mando *P* procedente del calculador. La flecha de doble trazo indica el sentido de la transferencia. En el ejemplo se transfiere el contenido del registro A al registro *B*. Examinemos ahora los diferentes casos de transferencia paralela entre dos registros *A* y *B*.

"La transferencia se efectuará coincidiendo con un impulso de transferencia Trf".

4.2 Transferencias paralelas. Utilización de las básculas R-S

4.2.1 *Transferencia precedida de una puesta a cero*

Se trata, en este caso, de transferir bajo el mando *P* del calculador, el contenido del registro *A* al registro *B*, es decir se realiza la operación:

$$P : (A) \Rightarrow B .$$

"El registro B ha sido previamente puesto a cero" por un impulso sobre la entrada *R* de las básculas *R-S*. La transferencia tiene lugar coincidiendo con un impulso de transferencia Trf. El cuadro de certeza de la operación de transferencia está representado en la figura 77. A_{it}, B_{it} representan el estado de una báscula del registro *A* y *B* (i-ésimo bit) en el instante *t*. En el instante $t + 1$ después de la aparición del impulso de transferencia, el estado de la báscula B_1 es $B_{i(t+1)}$. La figura 78 representa la realización del circuito de transferencia de $(A) \Rightarrow B$ para dos registros de tres pesos. La lectura del cuadro de certeza, atribuyendo a $\emptyset$ el valor cero, da:

$$R_i = 0$$

$$S_i = A_i \overline{B_i} P \,\mathrm{Trf} = A_i P \,\mathrm{Trf} \quad (B_i = 0) .$$

P	Trf	A_{it}	B_{it}	$B_{i(t+1)}$	R_i	S_i
1	1	0	0	0	$\emptyset$	0
1	1	1	0	1	0	1

Fig. 77. *Cuadro de certeza de (A) ⇒ B puesta a cero previa de B*

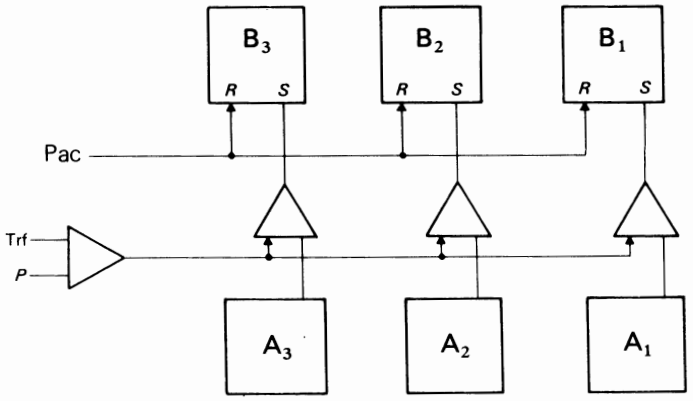


Fig. 78. *Transferencia en paralelo de (A) ⇒ B con puesta a cero previa del registro B*

4.2.2 *Transferencia sin puesta a cero previa*

Se desea transferir el contenido del registro A al registro B cualquiera que sea el contenido de éste. *"No hay puesta a cero previa"* del registro B. Las básculas utilizadas son del tipo R-S. Como anteriormente, se realiza la operación:

$$P : (A) \Rightarrow B .$$

El cuadro de certeza está representado en la figura 79.

P	Trf	A_{it}	B_{it}	$B_{i(t+1)}$	R_i	S_i
1	1	0	0	0	∅	0
1	1	1	0	1	0	1
1	1	0	1	0	1	0
1	1	1	1	1	0	∅

Fig. 79. *Cuadro de certeza de (A) ⇒ B sin puesta a cero previa de B*

La lectura de este cuadro, tomando para ∅ el valor 1, da:

$$R_i = \overline{A}_i(B_i + \overline{B}_i)\, P\, \mathrm{Trf} = \overline{A}_i\, P\, \mathrm{Trf}$$
$$S_i = A_i(B_i + \overline{B}_i)\, P\, \mathrm{Trf} = A_i\, P\, \mathrm{Trf}$$

de donde se obtiene la realización de la figura 80. Igualmente se pueden utilizar los circuitos "Y" de tres entradas *AP* Trf o $\overline{AP}$ Trf sobre cada entrada *R* y *S* de cada báscula, el resultado es idéntico al de la figura 80.

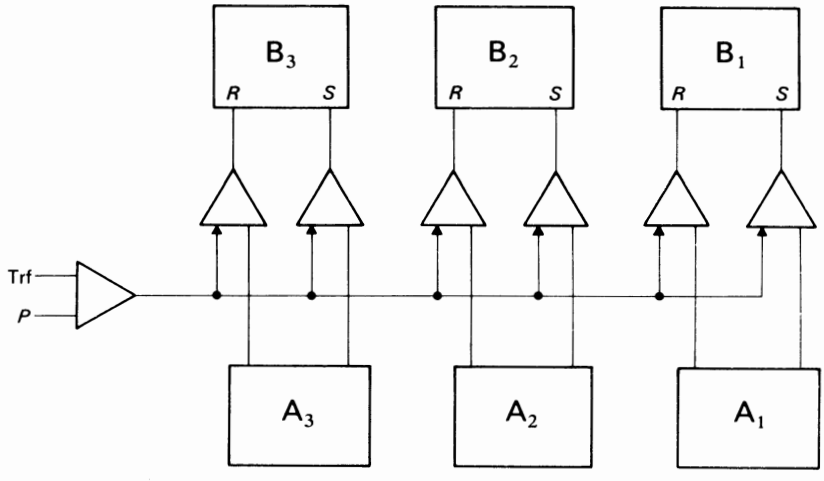


Fig. 80. *Transferencia en paralelo de (A) ⇒ B sin puesta a cero previa de B*

4.2.3 *Transferencia paralela del complemento a* 1 *con puesta a cero previa*

Se trata de transferir el complemento del contenido de *A* al registro *B* controlado por el mando *P* del ordenador coincidiendo con un impulso de transferencia Trf, sea:

$$P : (\overline{A}) \Rightarrow B .$$

El cuadro de certeza está representado en la figura 81. Haciendo $\emptyset = 0$ tenemos:

$$R_i = 0 ,$$

$$S_i = \overline{A}_i \, \overline{B}_i \, P \, \mathrm{Trf} = \overline{A}_i \, P \, \mathrm{Trf} \quad (B_i = 0) .$$

P	Trf	A_{it}	B_{it}	$B_{i(t+1)}$	R_i	S_i
1	1	0	0	1	0	1
1	1	1	0	0	0	$\emptyset$

Fig. 81. *Cuadro de certeza de* $(\overline{A}) \Rightarrow B$ *con puesta a cero previa*

La realización está representada en la figura 82.

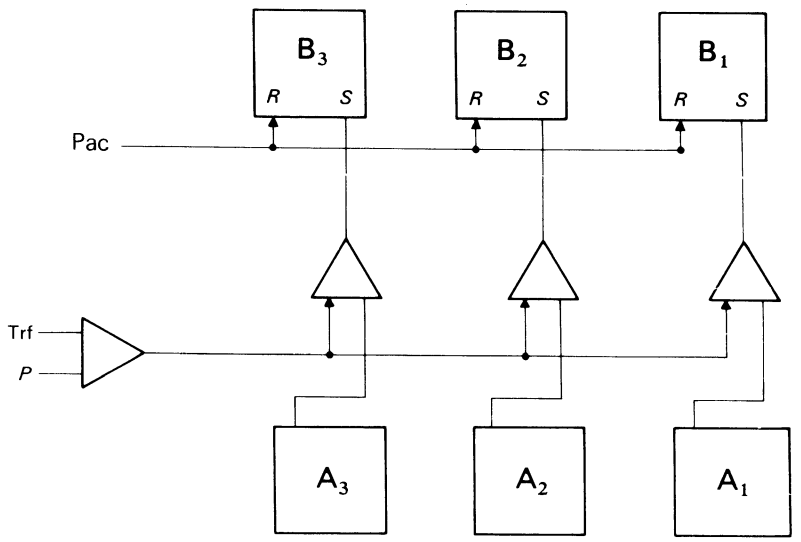


Fig. 82. *Transferencia en paralelo de* $(\overline{A}) \Rightarrow B$ *con puesta a cero previa del registro B*

4.2.4 *Transferencia del complemento a* 1 *sin puesta a cero previa*

Se desea obtener:

$$P : (\overline{A}) \Rightarrow B$$

cualquiera que sea el contenido de B: no hay puesta a cero previa de B. El cuadro de certeza está representado en la figura 83.

P	Trf	A_{it}	B_{it}	$B_{i(t+1)}$	R_i	S_i
1	1	0	0	1	0	1
1	1	0	1	1	0	$\emptyset$
1	1	1	0	0	$\emptyset$	0
1	1	1	1	0	1	0

Fig. 83. *Cuadro de certeza de* $(\overline{A}) \Rightarrow B$ *sin puesta a cero previa del registro B*

Haciendo $\emptyset = 1$ se obtiene:

$$R_i = A_i(B_i + \overline{B_i})\ P\ \mathrm{Trf} = A_i\ P\ \mathrm{Trf}\ ,$$
$$S_i = \overline{A_i}(B_i + \overline{B_i})\ P\ \mathrm{Trf} = \overline{A_i}\ P\ \mathrm{Trf}\ .$$

La realización del circuito está representada en la figura 84.

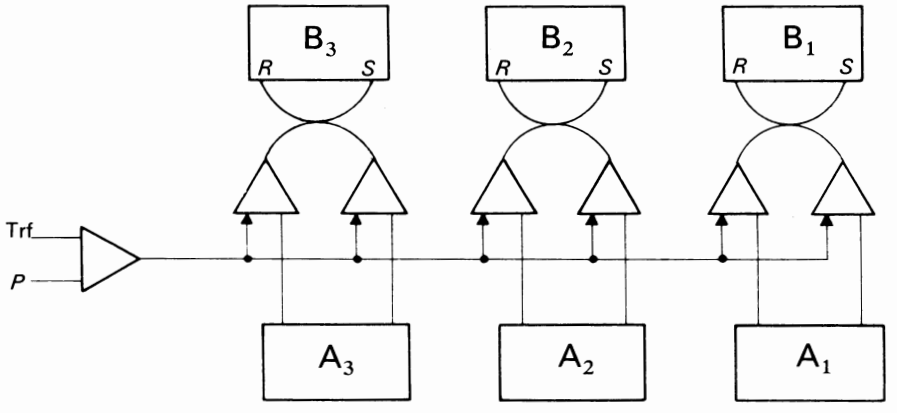


Fig. 84. *Transferencia de* $(\overline{A}) \Rightarrow B$ *sin puesta a cero previa de B*

4.3 Transferencia paralela por báscula T. (A) ⇒ B, sin puesta a cero previa del registro B

El registro B, en este caso, está constituido por básculas *T*. Refiriéndonos al cuadro de certeza de esta báscula, figura 27. El cuadro de certeza de este circuito está representado en la figura 85.

P	Trf	A_{it}	B_{it}	$B_{i(t+1)}$	T_i
1	1	0	0	0	0
1	1	0	1	0	1
1	1	1	0	1	1
1	1	1	1	1	0

Fig. 85. *Cuadro de certeza de (A) ⇒ B (básculas T)*

La lectura del cuadro de certeza da:

$$T_i = (\overline{A}_i B_i + A_i \overline{B}_i) P \text{ Trf} = (A_i \oplus B_i) \text{ P Trf} .$$

La realización está representada en la figura 86. El circuito "O exclusivo" será realizado con las salidas "verdad" y complementarias de las básculas de los registros *A* y *B*. Por razones de economía no se utilizarán circuitos "O exclusivo" separados.

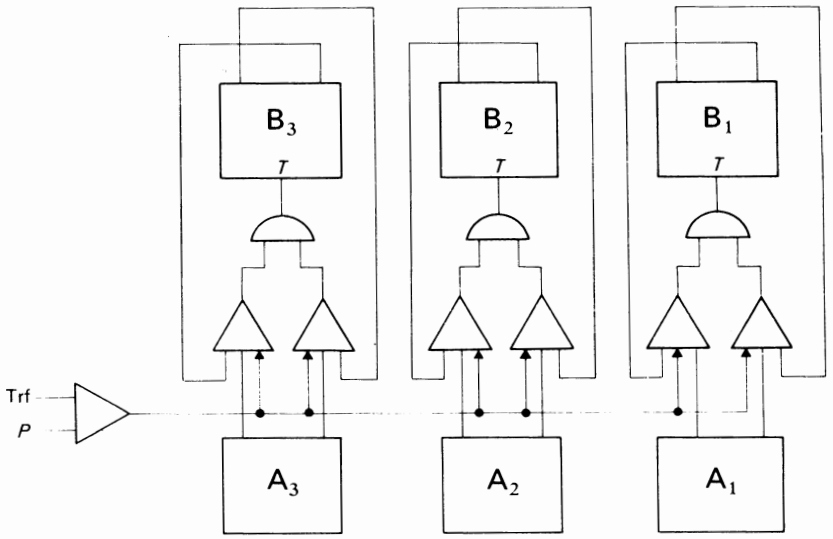


Fig. 86. *Transferencia en paralelo de (A) ⇒ (Básculas T)*

4.4 Adición lógica entre dos registros paralelos

En este caso se desea efectuar la suma lógica del contenido de dos registros A y B bajo el mando P del ordenador y coincidiendo con el impulso lógico Add log. Consignando el resultado en el registro B sea:

$$P : (A) + (B) \Rightarrow B .$$

Las básculas utilizadas son básculas R-S. El cuadro de certeza está representado en la figura 87. El registro B, no se pone a cero previamente.

P	Add log	$A_{i(t)}$	$B_{i(t)}$	$B_{i(t+1)}$	R_i	S_i
1	1	0	0	0	∅	0
1	1	0	1	1	0	∅
1	1	1	0	1	0	1
1	1	1	1	1	0	∅

Fig. 87. *Cuadro de certeza de (A) + (B) ⇒ B*

Los diagramas de Karnaugh de R_i y S_i están representados en la figura 88.

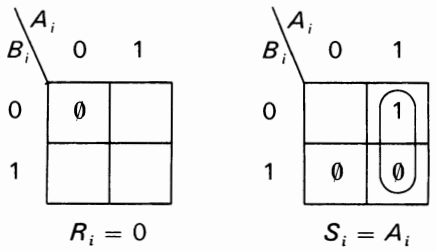


Fig. 88. *Diagramas de Karnaugh de (A) + (B) ⇒ B*

La lectura de los diagramas da:

$$R_i = 0 ,$$

$$S_i = A_i \; P \text{ Add log} .$$

La realización del circuito está representada en la figura 89.

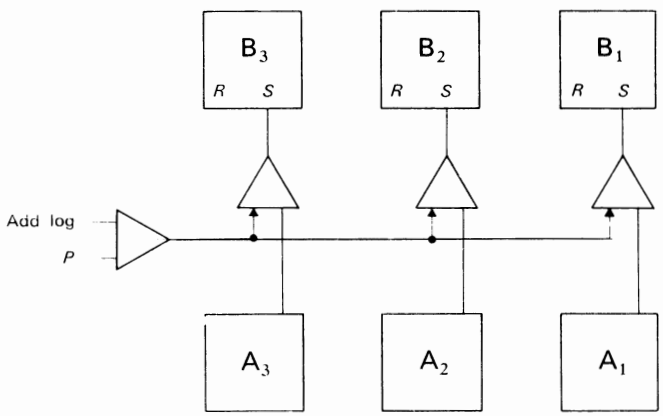


Fig. 89. *Realización de la operación (A) + (B) ⇒ B*

4.5 Multiplicación lógica entre dos registros paralelos

A veces es necesario efectuar la multiplicación lógica peso por peso del contenido de dos registros A y B. Realizándose la operación

$$P : (A).(B) \Rightarrow B .$$

El resultado tiene lugar bajo la solicitud del mando P y coincidiendo con un impulso Mult log, registrándose en B. El cuadro de certeza está representado en la figura 90.

P	Mult log	A_{it}	B_{it}	$B_{i(t+1)}$	R_i	S_i
1	1	0	0	0	∅	0
1	1	0	1	0	1	0
1	1	1	0	0	∅	0
1	1	1	1	1	0	∅

Fig. 90. *Cuadro de certeza de la operación (A) . (B) ⇒ B*

No existe puesta a cero previa de B ya que su contenido debe ser multiplicado por el contenido de A. En la figura 91 se representan los diagramas de Karnaugh correspondientes.

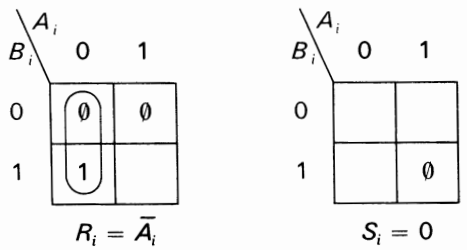


Fig. 91. *Diagramas de Karnaugh de la operación (A) . (B) ⇒ B*

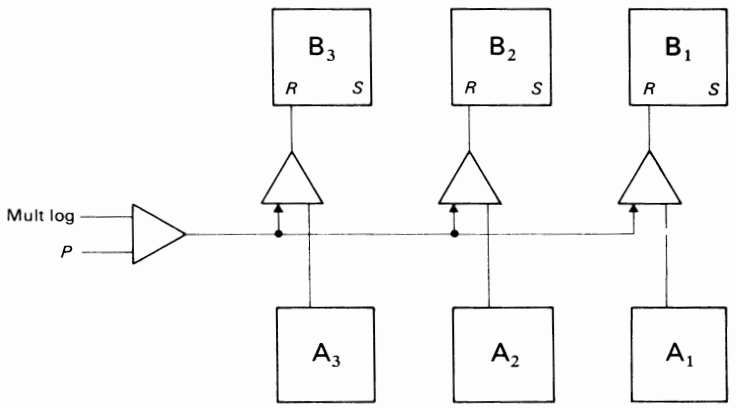


Fig. 92. *Realización de la operación (A) . (B) ⇒ B*

La lectura de los diagramas da:

$$R_i = \overline{A}_i \, P \text{ Mult log} ,$$

$$S_i = 0 .$$

El circuito de la realización está representado en la figura 92.

4.6 Suma "módulo 2" u "O exclusivo" entre dos registros

Se trata de realizar un "O exclusivo" peso por peso entre el contenido de dos registros A y B bajo control del mando P y coincidiendo con un impulso "suma módulo 2". El resultado será registrado en B:

$$P : (A) \oplus (B) \Rightarrow B .$$

Hemos visto en el capítulo 1 (tomo I) que la báscula T realiza la función $Q \oplus T$ entre su estado anterior Q y el estado de su entrada T. Utilizaremos pues básculas T para el registro B. El cuadro de certeza está representado en la figura 93.

P	Suma módulo 2	$A_{i(t)}$	$B_{i(t)}$	$B_{i(t+1)}$	T_i
1	1	0	0	0	0
1	1	0	1	1	0
1	1	1	0	1	1
1	1	1	1	0	1

Fig. 93. *Cuadro de certeza de la operación (A) ⊕(B) ⇒ B*

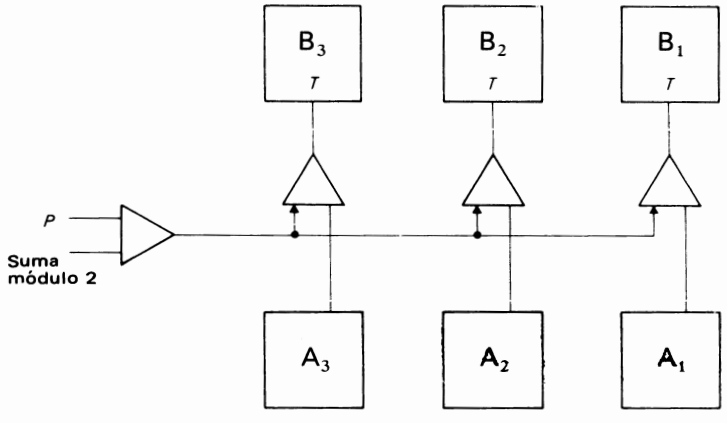


Fig. 94. *Realización de la operación (A) ⊕ (B) ⇒ B*

La lectura del cuadro de certeza da:

$$T_i = A_i \, P \text{ suma módulo 2.}$$

La realización está representada en la figura 94.

4.7 Transferencia paralela con conversión según el valor del signo

Es un problema de gran importancia que se presenta constantemente en la unidad aritmética cuando se desea realizar una sustracción. Se necesita convertir los números negativos en complemento a 1 (ó a 2 eventualmente). La transferencia de un registro a otro puede ser acompañada por la complementación del valor absoluto del número si su signo es 1 o puede ser transferido tal cual si su signo es cero. El signo puede ser conservado o igualmente ser complementado. El cuadro de certeza de la figura 93 corresponde a la conversión de un número negativo (signo 1), un número positivo (signo 0) se transfiere sin modificación.

P	Trf	Sgn	A_{it}	B_{it}	$B_{i(t+1)}$	R	S
1	1	0	0	0	0	∅	0
1	1	0	0	1	0	1	0
1	1	0	1	0	1	0	1
1	1	0	1	1	1	0	∅
1	1	1	0	0	1	0	1
1	1	1	0	1	1	0	∅
1	1	1	1	0	0	∅	0
1	1	1	1	1	0	1	0

Fig. 95. *Cuadro de certeza de (A) ⇒ B según Sgn*

Se realiza la operación siguiente

$$P : (A) \Rightarrow B \quad \text{si} \quad \text{Sgn } 0 \,,$$
$$P : (\overline{A}) \Rightarrow B \quad \text{si} \quad \text{Sgn } 1 \,.$$

Tomando para ∅ el valor ∅ = 1, obtendremos:

$$R_i = (A_i \odot \text{Sgn}) \, P \, \text{Trf} \,,$$
$$S_i = (A_i \oplus \text{Sgn}) \, P \, \text{Trf} \,.$$

El esquema de la realización está representado en la figura 96.

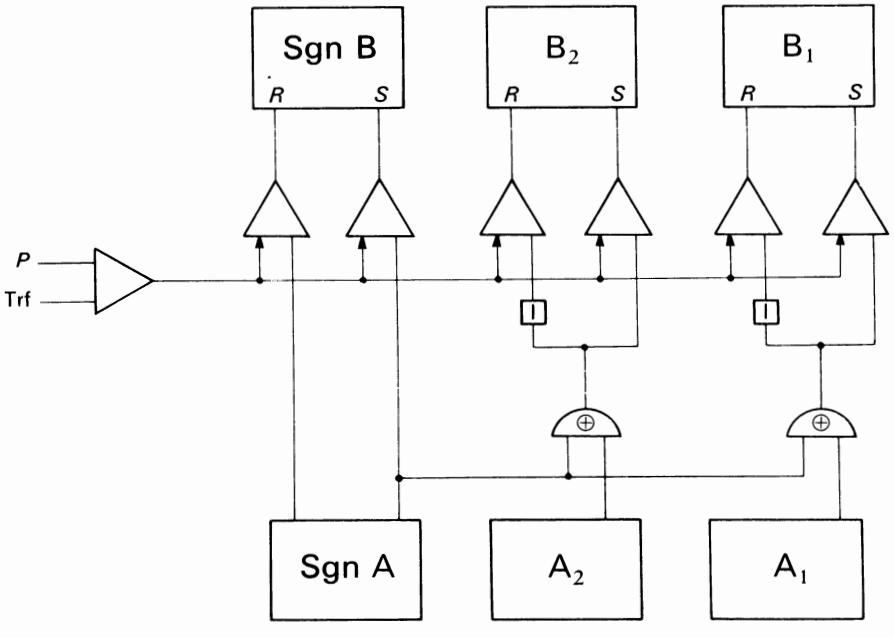


Fig. 96. *Transferencia con complemento del valor absoluto si el signo es negativo. El signo se conserva*

4.8 Transferencia en serie de una información de un registro a otro

En el caso de una transferencia en serie, la información contenida en un registro se transfiere en serie a otro registro. La transferencia puede ser o no precedida de una puesta a cero del registro receptor. Los registros que se utilizan son registros de desplazamientos y las conexiones entre etapas son idénticas.

La transferencia puede igualmente ser acompañada de una operación lógica: adición lógica, multiplicación lógica, suma módulo 2. Frecuentemente se desea efectuar la adición lógica entre dos registros e introducir el resultado en uno de ellos.

4.8.1 Adición lógica y transferencia en serie del resultado

En este caso, uno de los datos está en el registro A, el otro en el registro B. Se desea efectuar la adición lógica peso por peso del contenido del registro A con el contenido del registro B, bajo control del mando P del calculador:

$$P : (A) + (B) \Rightarrow B .$$

El cuadro de certeza está representado en la figura 97. Q_t es el estado de la primera báscula del registro B:

P	Avc	Q_t	A_i	B_i	Q_{t+1}	R_i	S_i
1	1	0	0	0	0	∅	0
1	1	0	0	1	1	0	1
1	1	0	1	0	1	0	1
1	1	0	1	1	1	0	1
1	1	1	0	0	0	1	0
1	1	1	0	1	1	0	∅
1	1	1	1	0	1	0	∅
1	1	1	1	1	1	0	∅

Fig. 97. *Cuadro de certeza de la transferencia en serie de la suma lógica: (A) + (B) ⇒ B*

Los diagramas de Karnaugh están representados en la figura 98.

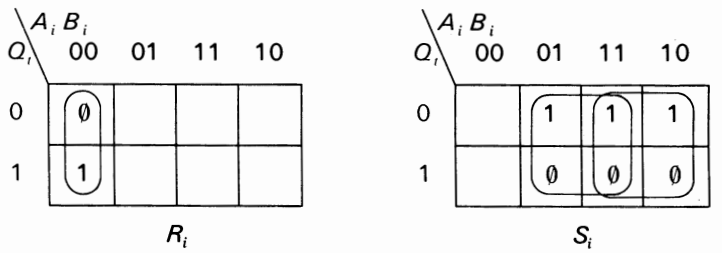


Fig. 98. *Diagrama de Karnaugh para la transferencia en serie de (A) + (B) ⇒ B*

La lectura de los diagramas da:

$$R_i = (\overline{A}_i\, \overline{B}_i)\, P\,\text{Avc}\,,$$

$$S_i = (A_i + B_i)\, P\,\text{Avc}\,.$$

El esquema de la realización está representado en la figura 99.

4.8.2 *Transferencia en serie de la multiplicación lógica*

Se trata de efectuar, bajo control del mando P del ordenador, la multiplicación lógica peso por peso del contenido de un registro A con el contenido del registro B. El resultado es transferido al registro B:

$$P : (A).(B) \Rightarrow B\,.$$

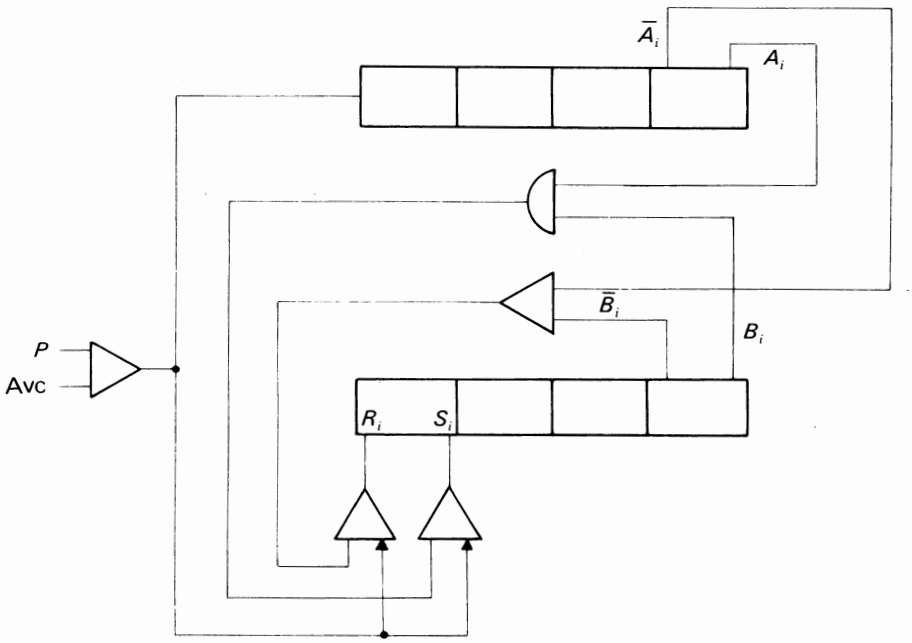


Fig. 99. *Realización de la transferencia en serie de (A) + (B) ⇒ B.*

El cuadro de certeza capaz para el i-ésimo peso de A y de B está representado en la figura 100.

P	Avc	Q_t	A_i	B_i	Q_{t+1}	R_i	S_i
1	1	0	0	0	0	∅	0
1	1	0	0	1	0	∅	0
1	1	0	1	0	0	∅	0
1	1	0	1	1	1	0	1
1	1	1	0	0	0	1	0
1	1	1	0	1	0	1	0
1	1	1	1	0	0	1	0
1	1	1	1	1	1	0	∅

Fig. 100. *Cuadro de certeza de la transferencia de la multiplicación lógica (A) . (B) ⇒ B*

Los diagramas de Karnaugh están representados en la figura 101. Las básculas de los dos registros son del tipo *R-S*.

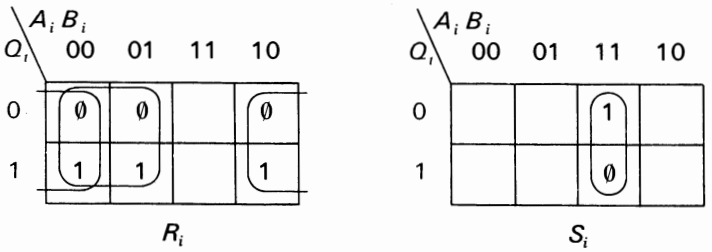


Fig. 101. *Diagrama de Karnaugh de la operación (A) . (B) ⇒ B*

La lectura de los diagramas de Karnaugh, coincidiendo con P y el avance, da:

$$S_i = (A_i\, B_i)\, P\,\text{Avc}\ ,$$

$$R_i = (\overline{A}_i + \overline{B}_i)\, P\,\text{Avc} = \overline{S}_i\, P\,\text{Avc}\ .$$

El esquema de la realización está representado en la figura 102. Obsérvese la posibilidad de utilizar un inversor para realizar $R_i = \overline{S_i}$.

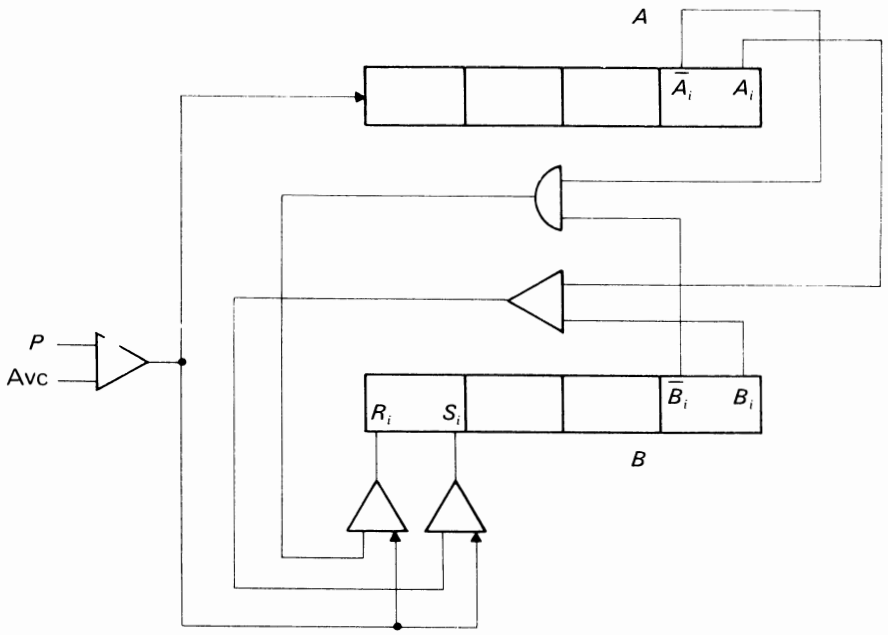


Fig. 102. *Realización de la transferencia en serie de (A) + (B) ⇒ B*

4.8.3 *Transferencia en serie de la suma sin arrastre (suma módulo 2)*

Se desea realizar, bajo control del mando P del ordenador, la suma sin arrastre peso por peso del contenido de un registro A con el contenido del registro B, es decir:

$$P : (A) \oplus (B) \Rightarrow B .$$

El resultado es transferido a B. Las básculas de los registros son del tipo *R-S*. El cuadro de certeza para los pesos A_i y B_i está representado en la figura 103. Los diagramas de Karnaugh correspondientes de R_i y S_i están representados en la figura 104.

P	Avc	Q_t	A_i	B_i	Q_{t+1}	R_i	S_i
1	1	0	0	0	0	∅	0
1	1	0	0	1	1	0	1
1	1	0	1	0	1	0	1
1	1	0	1	1	0	∅	0
1	1	1	0	0	0	1	0
1	1	1	0	1	1	0	∅
1	1	1	1	0	1	0	∅
1	1	1	1	1	0	1	0

Fig. 103. *Cuadro de certeza de la transferencia en serie de la suma (A) ⊕ B ⇒B*

La lectura de los diagramas, coincidiendo con P y el avance, da:

$$R_i = (A_i \odot B_i)\, P\, \text{Avc}$$

$$S_i = (A_i \oplus B_i)\, P\, \text{Avc} = \overline{R}_i\, P\, \text{Avc} .$$

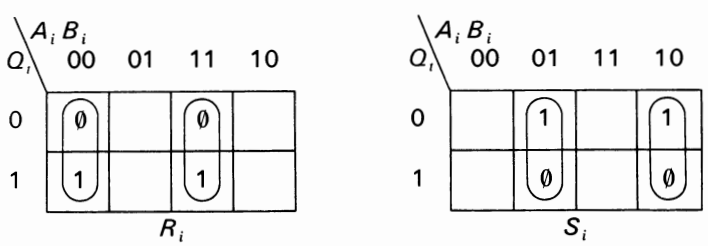


Fig. 104. *Diagramas de Karnaugh de la operación (A) ⊕ (B) ⇒B*

La realización está representada en la figura 105.

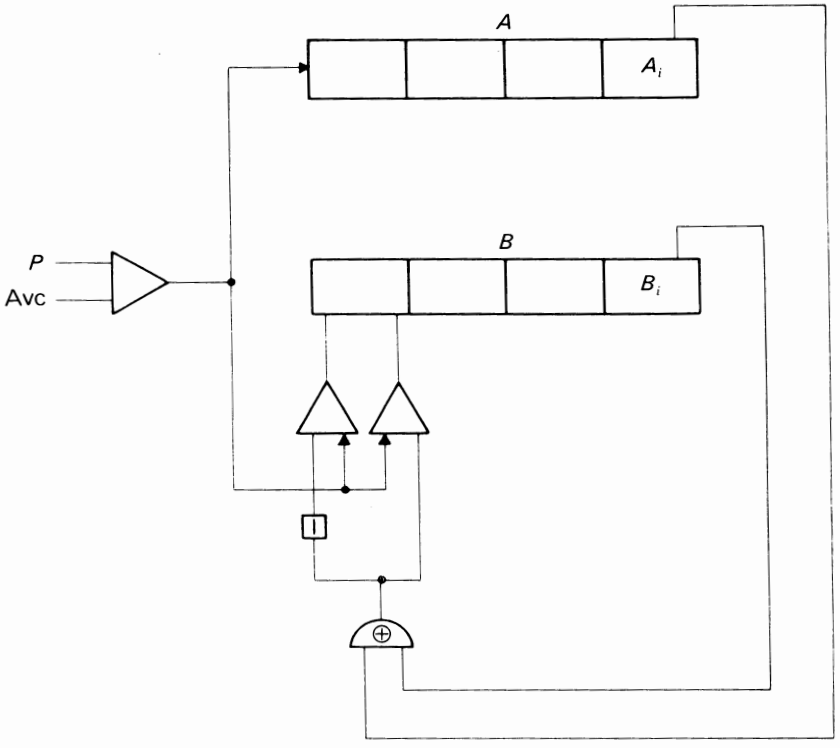


Fig. 105. *Realización de la transferencia en serie de (A) ⊕ (B) ⇒ B*

5. Los circuitos de complementación

En un calculador frecuentemente es necesario hacer uso de circuitos de complementación. Estos circuitos son necesarios cada vez que se desea realizar una sustracción. La naturaleza de los circuitos de complementación es diferente según que se desee hacer el complemento *a* 1, o el complemento *a* 2 de un número binario. Finalmente, según la naturaleza de las máquinas, se distinguen los circuitos en serie y en paralelo.

5.1 Circuito de complementación a 1

5.1.1 *Circuito en serie*

Se desea complementar peso por peso un número binario A, circulando en serie bajo control del mando P del calculador. Si $P = 1$ el número no se complementa, si $P = 0$ el número es complementado *a* 1. Recordemos que el complemento *a* 1

de un número binario equivale a cambiar los ceros en 1 y los 1 en ceros:

$$P : A_i \Rightarrow A_i \qquad (X_i = A_i) ,$$
$$P : A_i \Rightarrow \overline{A}_i \qquad (X_i = \overline{A}_i) .$$

El cuadro de certeza del circuito está representado en la figura 106. X es la función que realiza la complementación.

P	A_i	X_i
1	0	0
1	1	1
0	0	1
0	1	0

Fig. 106. *Cuadro de certeza del circuito de complementos a 1*

La lectura del cuadro de certeza da:

$$X_i = A_i P + \overline{A}_i \overline{P} = A_i \odot P .$$

El circuito tiene dos vías, una vía en la A_i no varía y una vía en la que A_i es complementado según el valor de P (Fig. 107).

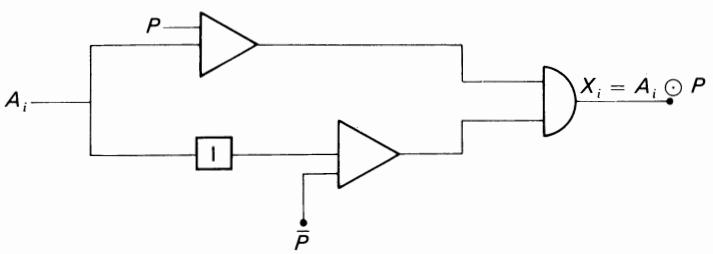


Fig. 107. *Circuito de complementos a 1 (circuito en serie)*

Si el número binario a complementar comprende n dígitos, se necesitarán n instantes elementales para obtener la complementación completa del número.

5.1.2 *Circuito en paralelo*

En este caso el número a complementar está en un registro A y la complementación se efectúa sobre todos los pesos en un solo instante elemental en el momento de realizar una transferencia a otro registro. Se necesitarán, pues,

tantos circuitos de complementación como cifras tenga el número. La función a realizar es:

$$X_i = A_i P + \overline{A}_i \overline{P} = A_i \odot P .$$

El esquema de la realización para un peso del número está representado en la figura 108.

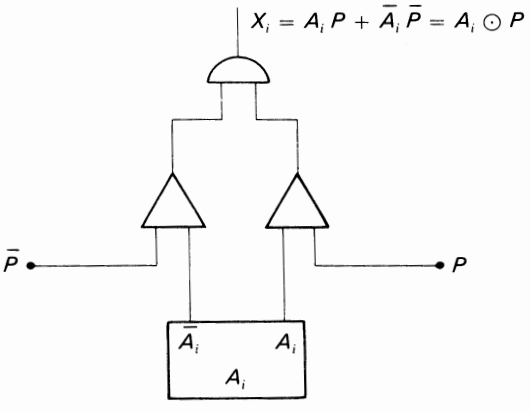


Fig. 108. *Circuito de complemento a 1 (circuito en paralelo) A_i es la báscula relativa al i-ésimo dígito del número a complementar*

5.2 Circuito de complemento a 2

Algunas máquinas utilizan el complemento *a* 2 para representar un número negativo. Recordemos que para complementar *a* 2 un número binario se cambian todos los 1 en 0 y los 0 en 1 añadiendo 1 sobre el último peso (el peso de menor rango decimal).

Ejemplo. *Sea el número* $A = 100110$. Su complemento *a* 2 será $\overline{A}_2$.

$$\begin{array}{rr} & 0\,1\,1\,0\,0\,1 \\ & + \qquad 1 \\ \overline{A}_2 & 0\,1\,1\,0\,1\,0 \end{array}$$

Comparemos peso por peso el número A y su complemento A_2 comenzando por los pesos de menor rango decimal:

$$\begin{aligned} A &= 1\,0\,0\,1\,1\,0 \\ \overline{A}_2 &= 0\,1\,1\,0\,1\,0 \end{aligned}$$

Todos los ceros del peso de menor rango decimal así como el primer 1 que sigue, no varían. Los demás dígitos son complementados *a* 1.

5.2.1 *Circuito en serie*

El circuito de realización prevé dos vías en función del valor atribuido a P. Si $P = 1$ el número binario no se complementa. Si $P = 0$ es complementado A 2. De todos modos es necesario conservar el primer 1 de menor rango decimal: lo que se obtiene por medio de la báscula C. Cuando ésta toma el valor 1, los dígitos sucesivos pasan por un inversor y son complementados *a* 1. Unicamente no varían los ceros si los hay y el 1 de menor rango decimal. De este modo se realiza el complemento *a* 2 del número binario A. El esquema está representado en la figura 109.

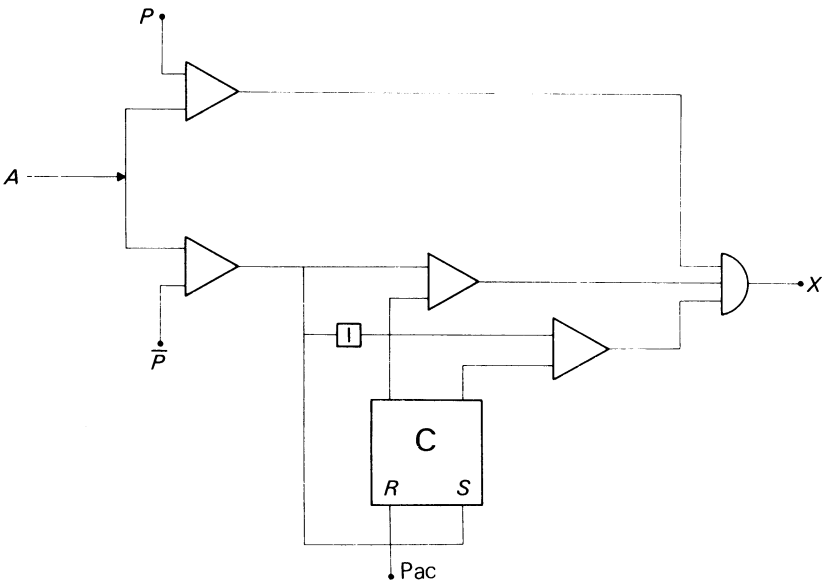


Fig. 109. *Circuito en serie de complemento a 2 de un número binario*

Más adelante veremos el caso de un circuito de complementación *a* 2 realizado con un adicionador en serie.

5.2.2 *Circuitos en serie para complementación a 2 de un número binario*

Se trata, en este caso, de efectuar el complemento *a* 2 cuando el número es negativo, es decir cuando su signo es igual a 1. El signo del número debe colocarse al principio de la secuencia de los 0 y los 1 del número, es decir, delante del dígito de menor rango decimal (menor peso). Se controla el signo. Si éste es positivo, el número se transfiere a la salida sin modificar. Si es negativo, se transfiere el signo a la salida sin modificar, pero el módulo del número es complementado *a* 2 de la misma forma que hemos visto anteriormente. El esquema de la realización

comprende un circuito idéntico al de la figura 109 precedido de un sistema de control de signo. Si éste es negativo, es tomado en memoria en una báscula signo que orienta los dígitos del módulo del número sobre el circuito de complementación.

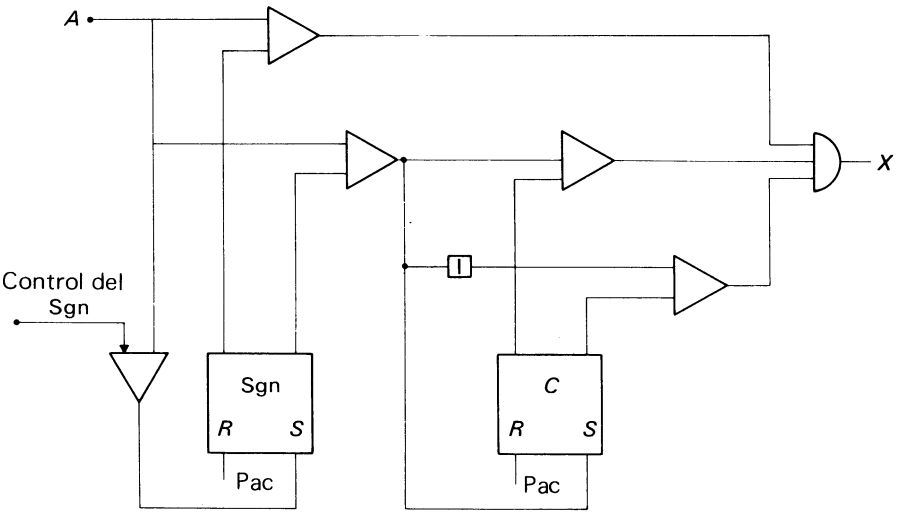


Fig. 110. *Circuito de complemento a 2 de un número binario con signo*

Capítulo 2

Operadores aritméticos

1. Diferentes tipos de adicionadores y sustractores

Pretendemos estudiar en este párrafo los diferentes tipos de adicionadores y sustractores de una máquina de calcular. Entendemos por adicionador y sustractor un circuito que realiza la suma y la diferencia de dos de los dígitos de un número teniendo en cuenta, eventualmente, los arrastres del orden inmediatamente inferior. Supongamos que se desea efectuar la suma de dos dígitos A_i y B_i de orden i y con un arrastre eventual C_{i-1} del orden inmediatamente inferior $i-1$. Generalmente son utilizados dos sistemas. La adición o sustracción se obtiene por medio de semiadicionadores o semisustractores, o por medio de adicionadores o sustractores completos. Finalmente, algunos circuitos combinan las dos funciones, es decir, realizan mediante un mando del calculador, la adición o sustracción de dos números A y B.

El resultado de la adición o la sustracción se introduce seguidamente en un dispositivo llamado *"acumulador"* y que estudiaremos más adelante. Examinemos ahora los diferentes tipos de adicionadores y de sustractores.

1.1 El semiadicionador

Es un dispositivo que realiza la adición de dos dígitos A_i y B_i sin arrastre del orden inmediatamente inferior. Suministra una suma sin arrastre X_i y un arrastre separado C_i. El cuadro de certeza del semiadicionador está representado en la figura 1.

A_i	B_i	X_i	C_i
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Fig. 1. *Cuadro de certeza del semiadicionador*

La única condición de existencia de un arrastre $C_i = 1$ es $A_i = B_i = 1$. La lectura del cuadro de certeza da:

$$X_i = \overline{A}_i B_i + A_i \overline{B}_i = A_i \oplus B_i ,$$

$$C_i = A_i B_i .$$

La suma sin arrastre de los dos dígitos A_i y B_i corresponde al "O exclusivo" entre estos dos dígitos y el arrastre C_i a la multiplicación lógica $A_i B_i$. El esquema de realización está representado en la figura 2 y el símbolo general de un semiadicionador en la figura 3.

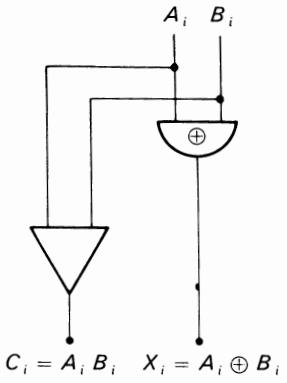


Fig. 2. *Realización del semiadicionador*

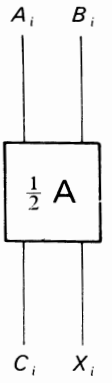


Fig. 3. *Símbolo del semiadicionador*

Más adelante examinaremos cómo se realiza un adicionador completo, es decir, teniendo en cuenta un arrastre eventual del orden inmediatamente inferior, por medio de dos semiadicionadores.

1.2 El semisustractor

En este caso, se trata de realizar la diferencia sin arrastre X_i de dos dígitos de orden i, A_i y B_i. El eventual arrastre C_i es tratado a parte en función de los valores de A_i y B_i. El cuadro de certeza del semisustractor está representado en la figura 4. Obsérvese que la condición de existencia de un arrastre $C_i = 1$ es $A_i = 0$ y $B_i = 1$.

A_i	B_i	X_i	C_i
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

Fig. 4. *Cuadro de certeza del semisustractor*

La lectura del cuadro de certeza, da:

$$X_i = A_i \oplus B_i ,$$
$$C_i = \overline{A}_i \, B_i .$$

También se observa que la diferencia sin arrastre X_i de dos dígitos A_i y B_i es idéntica a la suma sin arrastre de estos mismos dígitos. La sola diferencia entre las dos operaciones está en la expresión lógica del arrastre C_i y particularmente en el valor A_i que está invertido en el caso del semisustractor. La figura 5 representa el esquema de realización, la figura 6 el símbolo utilizado.

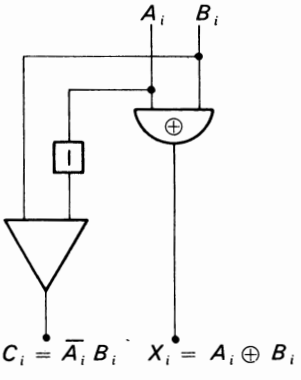


Fig. 5. *Realización del semisustractor*

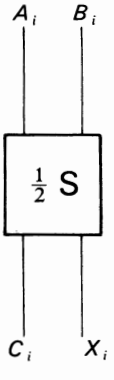


Fig. 6. *Símbolo del semisustractor*

1.3 Adicionador completo

Se entiende por adicionador completo, un dispositivo que efectúa la adición de dos dígitos A_i y B_i de orden i y de un arrastre eventual C_{i-1} de orden $i-1$ inmediatamente inferior. El cuadro de certeza está representado en la figura 7. El resultado de la adición es la nueva suma X_i y un arrastre eventual C_i hacia el orden superior. En la figura 8 se han representado los diagramas de Karnaugh de X_i y C_i. La lectura de estos diagramas da:

$$X_i = A_i \oplus B_i \oplus C_{i-1} ,$$
$$C_i = A_i \, B_i + C_{i-1}(A_i \oplus B_i) .$$

Suponemos, en este caso, que disponemos de un circuito "O exclusivo" para realizar la función $A_i \oplus B_i$. A pesar de que el circuito "O exclusivo" no sea considerado como una función elemental, los fabricantes de máquinas disponen siempre de circuitos que realizan la función $A_i \oplus B_i$, con el fin de simplificar los circuitos de hardware de los ordenadores. Es por esta razón que no hemos

A_i	B_i	C_{i-1}	X_i	C_i
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Fig. 7. *Cuadro de certeza del adicionador completo*

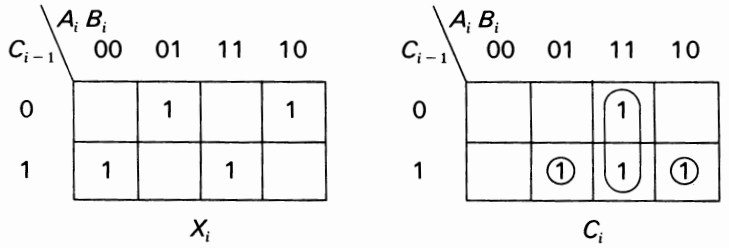


Fig. 8. *Diagramas de Karnaugh del adicionador completo*

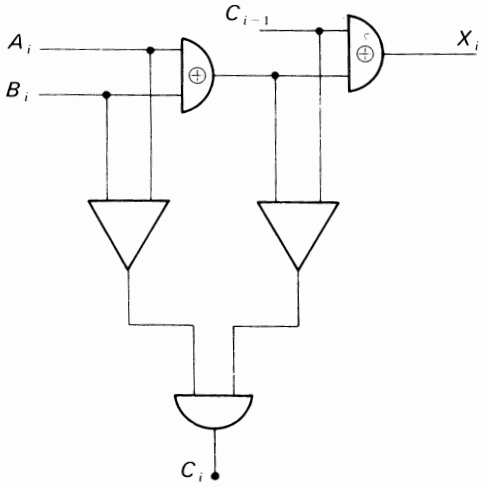


Fig. 9. *Realización del adicionador completo*

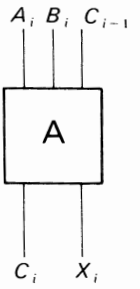


Fig. 10. *Símbolo del adicionador completo*

simplificado totalmente C_i, para poder beneficiarnos de la función $A_i \oplus B_i$ ya realizada en X_i. El esquema del adicionador completo está representado en la figura 9 y su símbolo en la figura 10.

Obsérvese que para realizar un adicionador completo basta con asociar dos semiadicionadores y reunir en un circuito "O" los arrastres eventuales, ya que éstos no pueden producirse simultáneamente. En el caso de una simplificación completa del diagrama de Karnaugh para el valor de C obtenemos:

$$C_i = A_i B_i + A_i C_{i-1} + B_i C_{i-1} ,$$

que es otra representación lógica del arrastre.

1.4 Sustractor completo

Se entiende por sustractor completo, un dispositivo que efectúa la diferencia entre dos dígitos A_i y B_i de orden i y de un arrastre C_{i-1} de orden $i-1$ inmediatamente inferior. El cuadro de certeza está representado en la figura 11. El resultado de la operación es la diferencia X_i y un arrastre eventual C_i. Se han representado en la figura 12 los diagramas de Karnaugh de X_i y C_i. La lectura de los diagramas da:

$$X_i = A_i \oplus B_i \oplus C_{i-1} ,$$

$$C_i = \overline{A}_i B_i + C_{i-1}(\overline{A_i \oplus B_i}) .$$

A_i	B_i	C_{i-1}	X_i	C_i
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

Fig. 11. *Cuadro de certeza del sustractor completo*

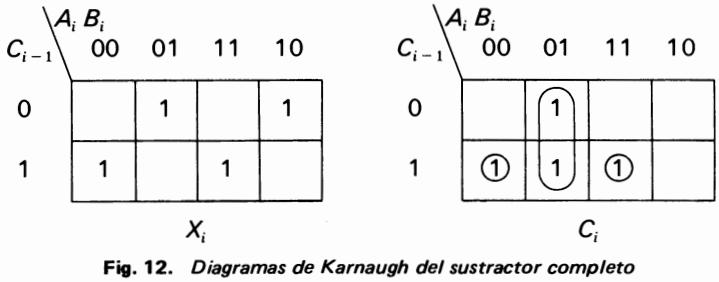


Fig. 12. *Diagramas de Karnaugh del sustractor completo*

Si se tiene en cuenta la solución simplificada de C_i, obtenemos para el valor del arrastre:

$$C_i = \overline{A}_i B_i + \overline{A}_i C_{i-1} + B_i C_{i-1} .$$

Suponemos en este caso, como anteriormente, que disponemos del circuito "O exclusivo" para realizar la función $A_i \oplus B_i$. Obsérvese que la diferencia X_i es idéntica a la suma X_i del adicionador completo. Para obtener el arrastre C_i basta con barrar la expresión A_i y $A_i \oplus B_i$ en el valor de C_i del adicionador completo. El esquema de realización y el símbolo de un sustractor completo están representados en la figura 13.

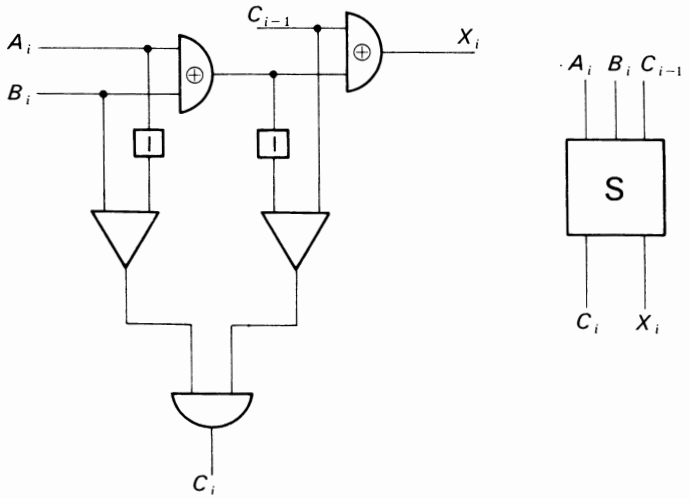


Fig. 13. *Realización y símbolo del sustractor completo*

Obsérvese que para realizar un sustractor completo basta con asociar dos semisustractores y reunir en un circuito "O" los arrastres eventuales, ya que éstos no se pueden producir simultáneamente.

1.5 Adicionador-sustractor

Se entiende por adicionador-sustractor, un dispositivo que efectúa, *bajo control del ordenador,* la suma o la diferencia de dos dígitos A_i y B_i de orden i y de un arrastre eventual C_{i-1} de orden $i-1$ de orden inmediatamente inferior. El adicionador-sustractor es, pues, la asociación de un adicionador y de un sustractor completos estudiados anteriormente. Dispone de dos mandos "Operation Addition" (Op Add) u "Operation Soustraction" (Op Sust),, que provocan su funcionamiento como adicionador o como sustractor. La figura 14 representa el esquema de un adicionador-sustractor así como su representación simbólica.

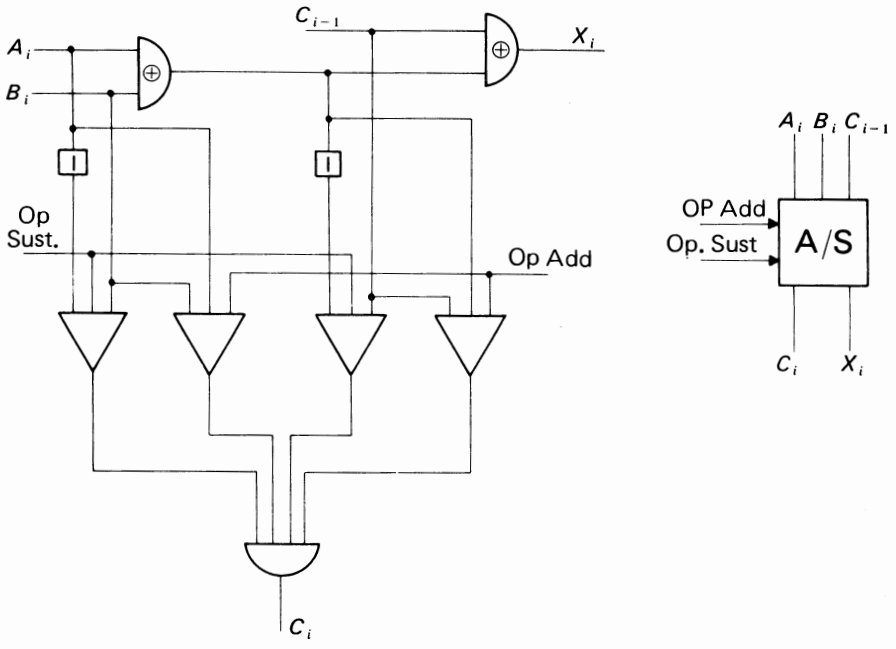


Fig. 14. *Realización y símbolo del adicionador-sustractor*

Para hacer una adición de mando Op Add es sensibilizado (Op Add = 1) y la salida C_i suministra un impulso de arrastre:

$$C_i = \{ A_i B_i + C_{i-1}(A_i \oplus B_i) \} \text{ Op Add} .$$

En el caso de una sustracción, es sensibilizado el mando Op Sust (Op Sust = 1) suministrando entonces la salida un impulso de arrastre:

$$C_i = \{ \overline{A}_i B_i + C_{i-1}(\overline{A_i \oplus B_i}) \} \text{ Op Sust}$$

En los dos casos, la suma o la diferencia es:

$$X_i = A_i \oplus B_i \oplus C_{i-1} .$$

Siendo pues independiente de los mandos Op Add y Op Sust.

1.6 Adicionador-sustractor que realiza las operaciones lógicas "Y" y "O exclusivo"

Hemos visto anteriormente que un semiadicionador suministra en la salida el "O exclusivo" de dos variables $X_i = A_i \oplus B_i$, así como el producto lógico. $C_i = A_i B_i$. Se utilizan estas propiedades para realizar con el adicionador-sustractor

las funciones lógicas "O exclusivo" y el producto lógico de dos variables. El adicionador-sustractor contendrá entonces cuatro mandos separados: adición, sustracción, multiplicación lógica, "O exclusivo". Sabemos que el "O exclusivo" de dos variables es la suma sin arrastres o la diferencia sin arrastres de estas dos variables. Así, para realizar $A_i \oplus B_i$ basta con suprimir los arrastres en el adicionador-sustractor. Para realizar A_iB_i se sensibilizará por medio de un mando Op Y, el circuito "Y" que suministra $C_i = A_i B_i$. Un mando Op $\oplus$ inhibe los arrastres eventuales. La modificación del esquema del adicionador-sustractor está representada en la figura 15.

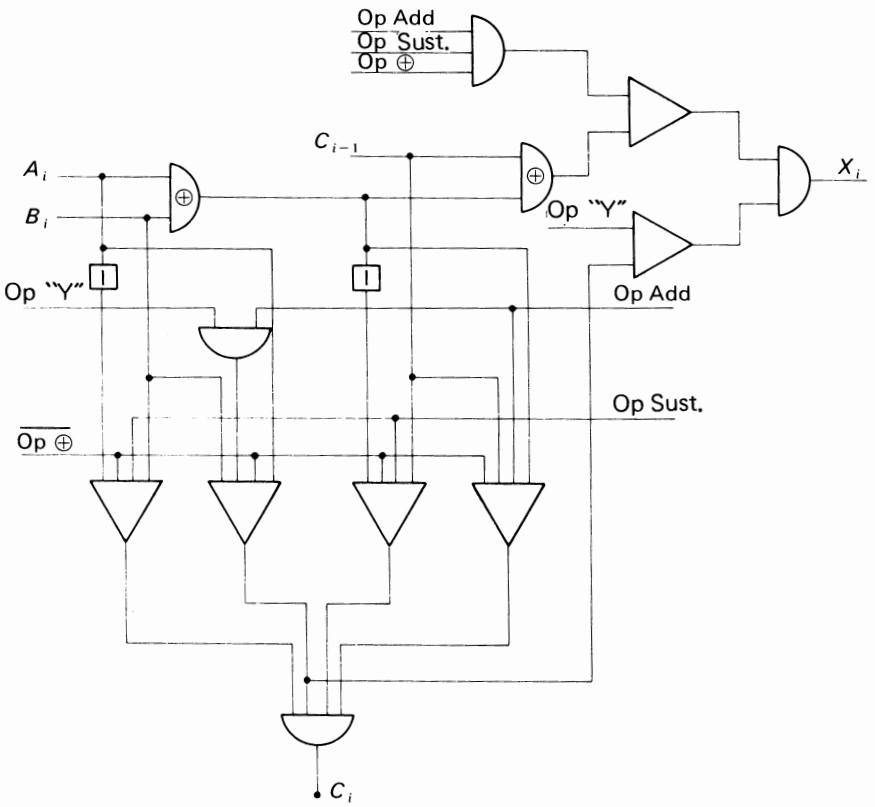


Fig. 15. *Adicionador-sustractor que efectúa las operaciones "Y" y ⊕. Etapa de orden i.*

1.7 Adicionador completo realizado únicamente con circuitos NOR y NAND

Un adicionador completo puede ser realizado únicamente con circuitos NOR o circuitos NAND. Modificaremos la ecuación lógica de C_i según el diagrama de Karnaugh de la figura 16.

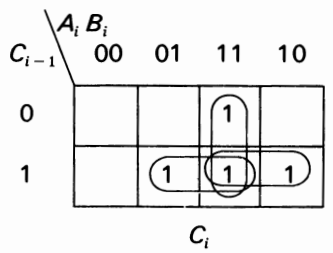


Fig. 16. *Diagramas de Karnaugh del arrastre C_i en el caso de una adición*

Con estas condiciones obtenemos:

$$C_i = A_i\, B_i + C_{i-1}(A_i + B_i)\,,$$

valor idéntico al encontrado anteriormente, pero resultante de un bucle diferente. La suma de los tres dígitos será:

$$X_i = A_i \oplus B_i \oplus C_{i-1}\ .$$

En la figura 17 se representa el esquema realizado utilizando circuitos NOR.

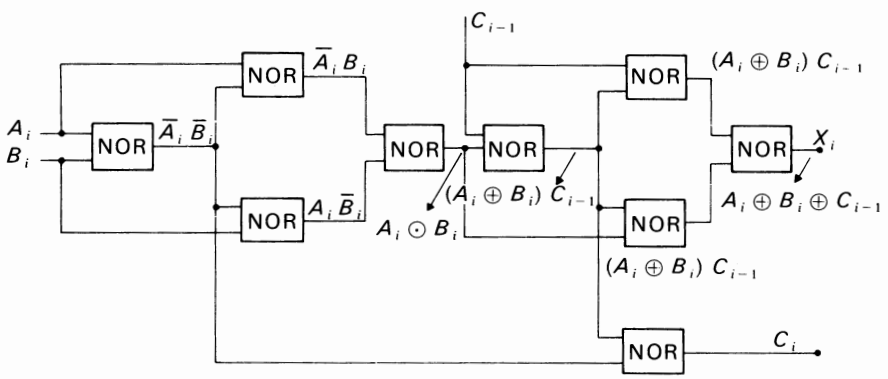


Fig. 17. *Adicionador completo realizado con circuitos NOR*

Si en el circuito anterior cambiamos los circuitos NOR por circuitos NAND, hay que transformar los productos en sumas. Obteniéndose de este modo la misma configuración que en la figura 17, pero cambiando los circuitos NOR por circuitos NAND (Fig. 18). Lo que proviene de la relación:

$$A_i \oplus B_i \oplus C_i = A_i \odot B_i \odot C_i\ .$$

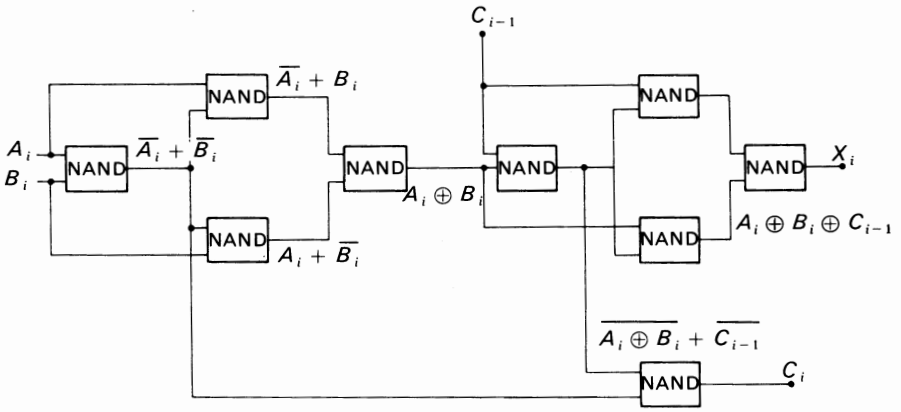


Fig. 18. *Adicionador completo realizado con circuitos NAND*

1.8 Adicionador-sustractor de básculas

Otro modo de realizar adicionadores-sustractores es utilizando dos registros A y B, que contienen los dos números a sumar, convenientemente conectados entre ellos. En efecto, la suma de dos números binarios A y B puede ser representada por una sucesión de sumas sin arrastres a las que se añaden los arrastres eventuales desplazados de una posición a la izquierda. El siguiente ejemplo ayudará a comprender mejor el mecanismo de la adición. Consideremos dos números 00111 contenidos en el registro A y 01011 en el registro D. La sucesión de las operaciones es la siguiente:

Registro A	0 0 1 1 1	
Registro D	0 1 0 1 1	
1ª suma sin arrastre	0 1 1 0 0	
Arrastres desplazados	0 0 1 1 0	
2ª suma sin arrastre	0 1 0 1 0	
Arrastres desplazados	0 1 0 0 0	
3ª suma sin arrastre	0 0 0 1 0	
Arrastres desplazados	1 0 0 0 0	
4ª suma sin arrastre	1 0 0 1 0	Resultado
Arrastres desplazados	0 0 0 0 0	

Cuando el registro D se ha quedado vacío, el resultado está en el registro A. Obsérvese que éste contiene la sucesión de sumas sin arrastre: haciendo las funciones de acumulador. Más adelante volveremos sobre este importante punto. Del mismo modo, el registro D contiene la sucesión de los arrastres. Al final de la operación su contenido es nulo. Para determinar las uniones entre las etapas de los

registros A y D nos referimos al cuadro de certeza de la figura 19 en la que A_{it}, D_{it}, $A_{i(t+1)}$ representan el estado de las básculas de orden i en el instante t y $t+1$ de los registros A y D. Igualmente $D_{(i-1),\,(t+1)}$ representa el estado de la báscula D de orden $i-1$ en el instante $t+1$.

A_{it}	D_{it}	$A_{i(t+1)} = A_{it} \oplus D_{it}$	$D_{i-1,\,t+1}$	R_{it}	S_{it}
0	0	0	0	∅	0
0	1	1	0	0	1
1	0	1	0	0	∅
①	1	⓪	①	1	0

Fig. 19. *Cuadro de certeza del adicionador de básculas de la figura 18*

Este tipo de adicionador funciona del modo siguiente: Una vez introducidos los dos números en A y D, encontrándose el número a adicionar en D, se pone a cero el registro D. Si la báscula D_i estaba a 1, así como la báscula del mismo orden del registro A_i, esta última deberá ser puesta a cero y memorizarse un arrastre para ser añadido al contenido de A_{i-1}. Como el registro D ha sido puesto a cero, toma en carga los arrastres haciendo la función de memoria de arrastres. Nuevamente se pone a cero el registro D, el registro A presenta la suma sin arrastre de su contenido más el contenido del registro D (arrastres desplazados). Los nuevos arrastres desplazados una posición a la izquierda son introducidos en el registro D, *cuando se hace una adición y que* A_i *pasa del valor* 1 *en el instante t, al valor* 0 *en el instante* $t+1$.

La lectura del cuadro de certeza, da:

$$R_i = A_{it}\, D_{it}\,,$$
$$S_i = \overline{A}_{it}\, D_{it}\,.$$

Estas ecuaciones lógicas son igualmente válidas en el caso de una sustracción, pero en este caso habrá arrastre hacia el orden superior de D cuando A_i en el instante t pasa del valor 0 al valor 1 en el instante $t+1$. Luego en los dos casos, basta con reunir las salidas "verdad" y las "falsas" de cada báscula A_i, con la condición adición y sustracción, a cada entrada S_i de cada báscula del registro D. El esquema general está representado en la figura 20. Obsérvese que la adición en los dos tipos de adicionador-sustractor anteriores tarda cierto tiempo en realizarse. Este tiempo depende de la naturaleza de los dos números y de los arrastres que hay que sumar sucesivamente a la suma sin arrastre de los dos números.

También hemos introducido en el esquema de la figura 20 las operaciones "Y" y "O exclusivo" entre el contenido de los dos registros A y D. La operación "Y" se efectúa en el tiempo elemental T.

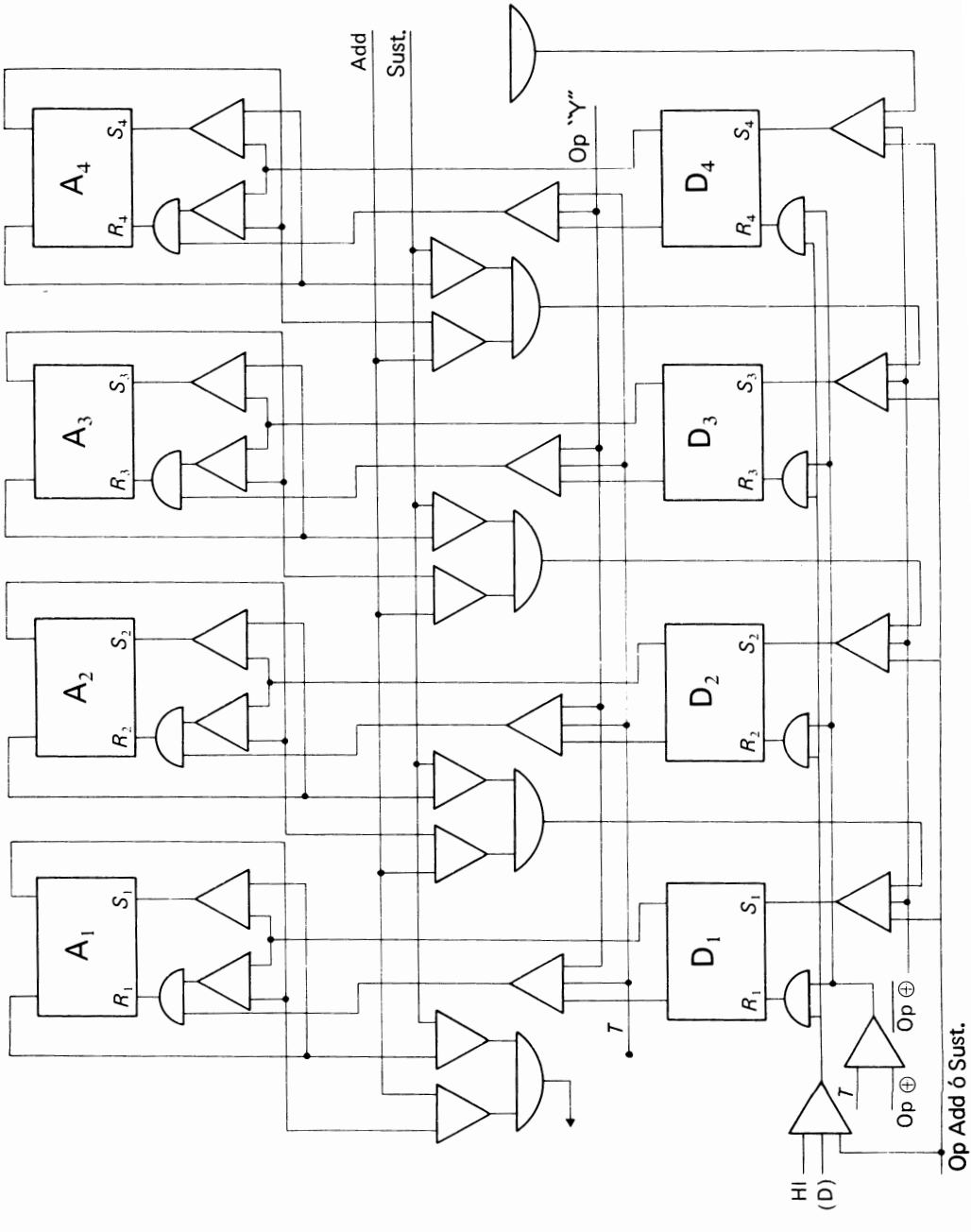


Fig. 20. *Adicionador-sustractor de básculas*

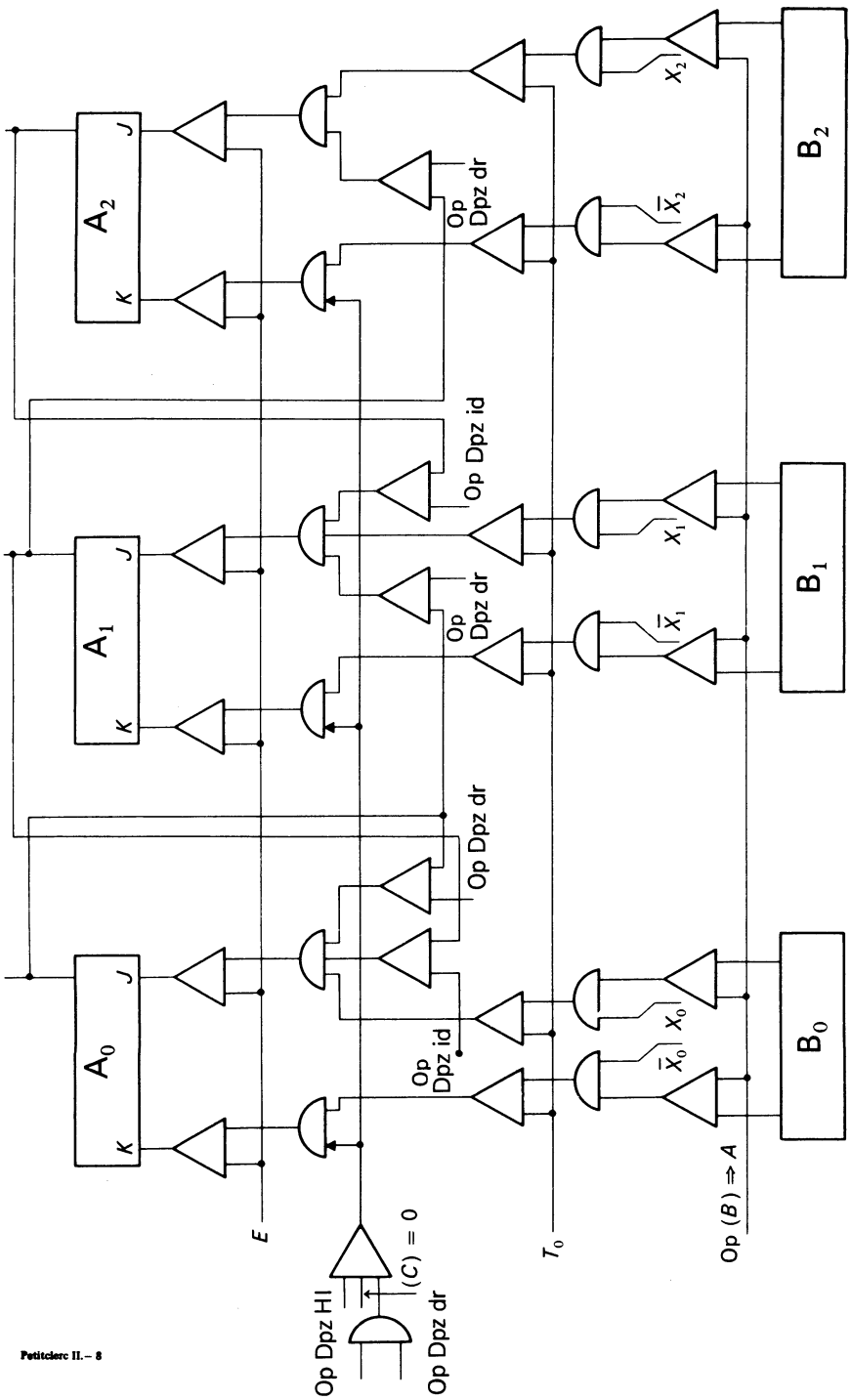


Fig. 21. *Ejemplo de realización de un acumulador*

1.9 El acumulador

Se entiende por acumulador, un registro que recibe, bajo una orden que envía el ordenador: la suma, la diferencia, las operaciones "Y", "O" y "O exclusivo", efectuadas por el adicionador, entre su contenido y el del registro R. Se puede, igualmente, con un mando exterior, desplazar su contenido un cierto número de posiciones a la derecha o a la izquierda. En el caso de un desplazamiento a la derecha se conserva el signo del resultado, es decir, el contenido del peso de mayor rango decimal. Finalmente, es posible igualmente, por medio de un mando exterior, transferir su contenido memorizado con la ayuda del registro R, y recibir el contenido de este mismo registro. La figura 21 representa el esquema de este acumulador. Las entradas X_i y $\overline{X_i}$ representan las salidas "verdad" y complementarias, del adicionador-sustractor de la figura 15. Obsérvese que todas las operaciones de transferencia se efectúan durante el ciclo de ejecución E en el tiempo T_0 de la máquina. Las operaciones de desplazamientos son controladas por el reloj (Hl). Se necesitarán tantos impulsos de reloj como posiciones a desplazar. La operación de desplazamiento termina cuando el contenido del contador de desplazamientos es nulo. Recuérdese que el número de desplazamientos a efectuar es introducido en el contador de desplazamientos al principio de la operación, disminuyendo este contador en una unidad a cada impulso del reloj.

2. Adicionador-sustractor en serie

Los adicionadores-sustractores que hemos estudiado hasta aquí son del tipo "en paralelo". Esto supone que los dígitos a adicionar llegan a las entradas al mismo tiempo y que la suma o la diferencia se obtiene *simultáneamente* en todos los pesos. Luego debe haber tantas etapas adicionadoras-sustractoras como dígitos en los números a tratar. El proceso es muy rápido pero bastante costoso desde el punto de vista circuitos. Otro método menos rápido, pero de concepto más simple, consiste en efectuar una adición o una sustracción en serie. En este caso los dígitos de los dos números no son adicionados o sustraídos simultáneamente, sino sucesivamente, dígito tras dígito, comenzando por los de menor rango decimal. En este sistema es suficiente con la utilización de un solo adicionador-sustractor. La figura 22 representa el esquema de un adicionador-sustractor en serie. Proporciona, como los anteriores, la suma o la diferencia de dos dígitos A_i y B_i y de un arrastre eventual C_{i-1} del orden $i-1$.

$$X_i = A_i \oplus B_i \oplus C_{i-1} .$$

Si la suma o la diferencia X_i da lugar a un arrastre C_i, éste se memoriza al producirse el impulso de avance siguiente, en una báscula de arrastres C, y se añade a los dos dígitos siguientes A_{i+1} y B_{i+1}. El arrastre no es suministrado por el adicionador-sustractor, éste es elaborado al mismo tiempo que la suma o la diferencia a partir de A_i y B_i.

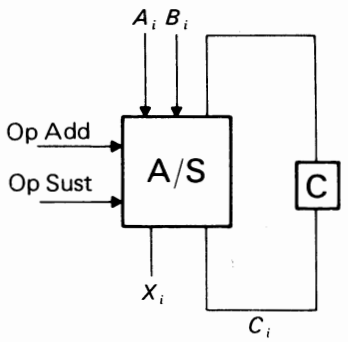


Fig. 22. *Esquema general de un adicionador-sustractor en serie*

2.1 Adicionador en serie

La figura 23 representa el cuadro de certeza del adicionador en serie. La primera columna es el impulso de avance. Las otras tres columnas reflejan las combinaciones de los valores A_i B_i y C_{i-1} presentes en la entrada del adicionador, las dos últimas columnas representan las condiciones que hay que aplicar sobre las entradas R y S de la báscula de arrastre C para que ésta pase del valor C_{i-1} en el instante t al valor C_i, al impulso de avance siguiente al instante $t+1$. Llevando estos valores a los diagramas de Karnaugh (Fig. 24) de R y de S obtenemos, después de simplificar:

$$R_i = \overline{A_i}\,\overline{B_i}\,\mathrm{Avc}\ ,$$

$$S_i = A_i\,B_i\,\mathrm{Avc}\ .$$

Avc	A_i	B_i	C_{i-1}	X_i	C_i	R_i	S_i
1	0	0	0	0	0	∅	0
1	0	0	1	1	0	1	0
1	0	1	0	1	0	∅	0
1	0	1	1	0	1	0	∅
1	1	0	0	1	0	∅	0
1	1	0	1	0	1	0	∅
1	1	1	0	0	1	0	1
1	1	1	1	1	1	0	∅

Fig. 23. *Cuadro de certeza del adicionador en serie*

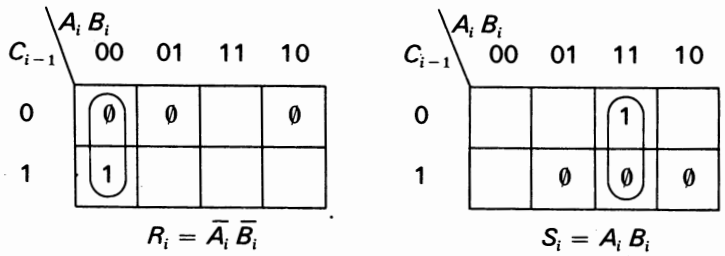


Fig. 24. *Diagramas de Karnaugh de la báscula de arrastre del adicionador en serie*

El esquema general del adicionador en serie está representado en la figura 25. Se supone que los dos números a adicionar A y B han sido previamente introducidos en serie en los registros respectivos. Obsérvese que cada dígito X_i de la suma se vuelve a introducir en el registro A que contiene la suma al final de la operación. Haciendo la función de acumulador. Por el contrario, el contenido del registro B se pierde.

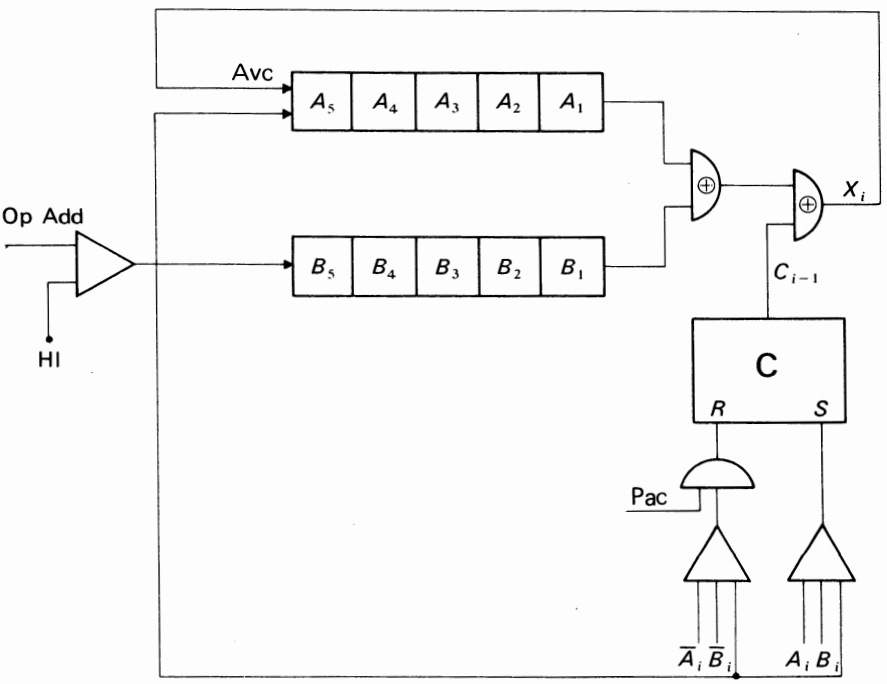


Fig. 25. *Esquema de principio del adicionador en serie*

2.2 Sustractor en serie

La figura 26 representa el cuadro de certeza del sustractor en serie. Así como los nuevos valores de R y S de la báscula de arrastre cuando se efectúa la sustracción de A_i y de B_i. Llevando estos valores a los diagramas de Karnaugh de R y de S (Fig. 27), obtenemos, después de simplificar:

$$R_i = A_i \, \overline{B}_i \, \text{Avc} \, ,$$

$$S_i = \overline{A}_i \, B_i \, \text{Avc} \, .$$

Avt	A_i	B_i	C_{i-1}	X_i	C_i	R_i	S_i
1	0	0	0	0	0	∅	0
1	0	0	1	1	1	0	∅
1	0	1	0	1	1	0	1
1	0	1	1	0	1	0	∅
1	1	0	0	1	0	∅	0
1	1	0	1	0	0	1	0
1	1	1	0	0	0	∅	0
1	1	1	1	1	1	0	∅

Fig. 26. *Cuadro de certeza del sustractor en serie*

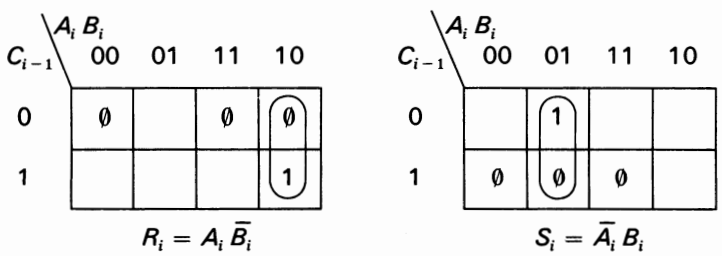


Fig. 27. *Diagramas de Karnaugh de la báscula de arrastre del sustractor en serie*

La figura 28 representa el esquema de principio del sustractor en serie que solamente se diferencia del adicionador por las condiciones aplicadas sobre las entradas R y S de la báscula de arrastre.

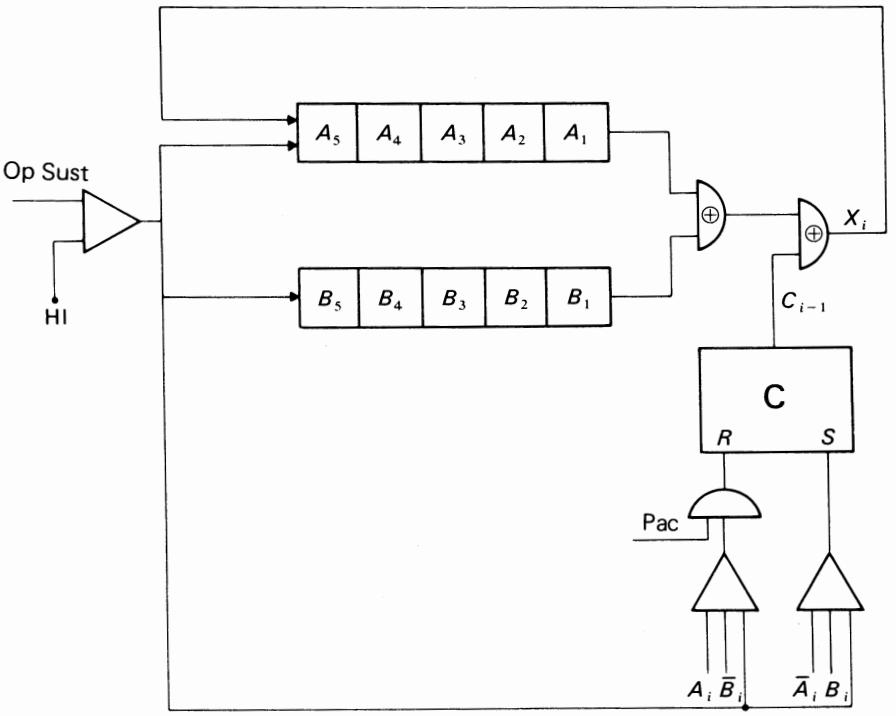


Fig. 28. *Esquema de principio del sustractor en serie*

2.3 Adicionador-sustractor

Las dos modalidades de funcionamiento anteriores pueden ser reunidas en una sola. Para ello es necesario prever dos entradas suplementarias Op Add y Op Sust con el fin de determinar la operación a realizar (adición o sustracción). En este caso se realizarán las operaciones lógicas siguientes:

$$X_i \Rightarrow A_5 = (A_i \oplus B_i \oplus C_{i-1}) \text{ Avc} ,$$

$$R_i = (\overline{A}_i \, \overline{B}_i \text{ Op Add} + A_i \, \overline{B}_i \text{ Op Sust}) \text{ Avc} ,$$

$$S_i = (A_i \, B_i \text{ Op Add} + \overline{A}_i \, B_i \text{ Op Sust}) \text{ Avc} .$$

La primera de estas expresiones significa que cada dígito de la suma o la diferencia X_i es introducido en A_5 coincidiendo con la recaída del impulso de avance. El esquema de principio del adicionador-sustractor está representado en la figura 29. También aquí el registro A hace la función de acumulador. Al final de la operación contendrá la suma o la diferencia de dos números A y B. El contenido de B se pierde como en el caso anterior.

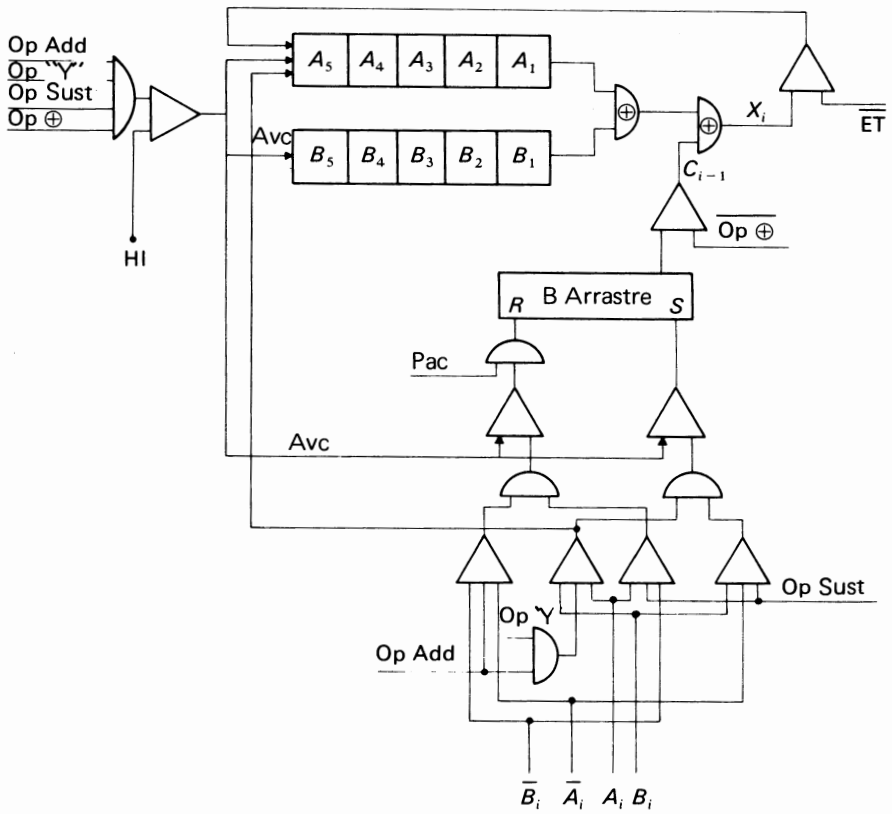


Fig. 29. *Esquema de principio del adicionador-sustractor en serie*

3. Adicionadores decimales

Se entiende por adicionador decimal, un dispositivo que efectúa la suma de dos dígitos decimales. Como cada dígito decimal está representado por cuatro dígitos binarios, se trata pues de un dispositivo que posee cuatro entradas dobles: una para cada dígito decimal, y una entrada suplementaria para un arrastre eventual procedente del orden inmediatamente inferior. Este circuito suministra la suma decimal de dos dígitos. Comprende igualmente cuatro salidas para la suma y una salida suplementaria para un arrastre eventual. La representación de cada dígito decimal por cuatro dígitos binarios recibe el nombre de "*Tetrada*". Luego para efectuar la suma de dos cifras decimales, basta con adicionar sus tetradas respectivas. Obsérvese que con una tetrada (cuatro dígitos binarios) se pueden representar los números decimales comprendidos entre 0 y 15, de los cuales solamente 10 nos interesan, los comprendidos entre 0 y 9. Las seis combinaciones

correspondientes a los valores decimales de 10 a 15 no se utilizan. Estas combinaciones no válidas reciben el nombre de seudotetradas. Dichas seudotetradas pueden aparecer en el curso de la adición de dos tetradas dando un resultado falso. Es pues necesario detectarlas para corrección. El cuadro de la figura 31 representa las tetradas y seudotetradas en el código *BCD*.

Estas seudotetradas se corrigen por medio de un circuito corrector que proporciona el resultado exacto.

Obsérvese que cuando el adicionador suministra un arrastre en el orden más elevado de la tetrada, es decir, cuando el adicionador suma los números 8 y 9 por ejemplo, *a pesar de no ser una seudotetrada, el resultado debe ser igualmente corregido*. Según lo estudiado en el capítulo 3 del tomo I, basta con añadir $6_{10} = 0110$ al resultado del adicionador para obtener el resultado exacto. Esta corrección se efectuará cuando se detecte un arrastre más allá del orden más elevado (adición de 8 y 9 por ejemplo), o cuando el resultado de la adición sea una seudotetrada. En la figura 32 se representa el esquema del adicionador decimal. El circuito correcto no tiene adicionador para el orden más bajo ya que siempre se añade 0, ni tampoco arrastre más allá del orden más elevado porque éste se elabora en el momento de la detección de una seudotetrada o de una adición de dos números decimales superiores a 8. Llamemos P a la salida del circuito detector de las seudotetradas. Refiriéndonos al cuadro de la figura 31 obtendremos el diagrama de Karnaugh de la figura 30.

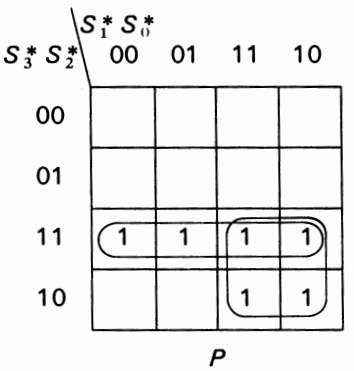


Fig. 30. *Diagrama de Karnaugh del detector de seudotetradas en el código BCD*

La lectura del diagrama, da:

$$P = S_2^* S_3^* + S_1^* S_3^* = S_3^*(S_1^* + S_2^*),$$

de donde se obtiene el circuito de detección de la figura 33. Con este tipo de adicionador es pues posible realizar un adicionador completo. Presentándose éste bajo la forma de adicionador en paralelo o adicionador en paralelo y serie.

Decimal	Código *BCD* $S_3^* S_2^* S_1^* S_0^*$	Decimal	Seudo-tetradas $S_3^* S_2^* S_1^* S_0^*$	Código *B C D* $S_3^* S_2^* S_1^* S_0^*$
0	0 0 0 0	10	1 0 1 0	0 0 0 ① 0 0 0 0
1	0 0 0 1	11	1 0 1 1	0 0 0 ① 0 0 0 1
2	0 0 1 0	12	1 1 0 0	0 0 0 ① 0 0 1 0
3	0 0 1 1	13	1 1 0 1	0 0 0 ① 0 0 1 1
4	0 1 0 0	14	1 1 1 0	0 0 0 ① 0 1 0 0
5	0 1 0 1	15	1 1 1 1	0 0 0 ① 0 1 0 1
6	0 1 1 0	16	① 0 0 0 0	0 0 0 ① 0 1 1 0
7	0 1 1 1			
8	1 0 0 0			
9	1 0 0 1			

Fig. 31. *Tetradas y seudotetradas en el código BCD*

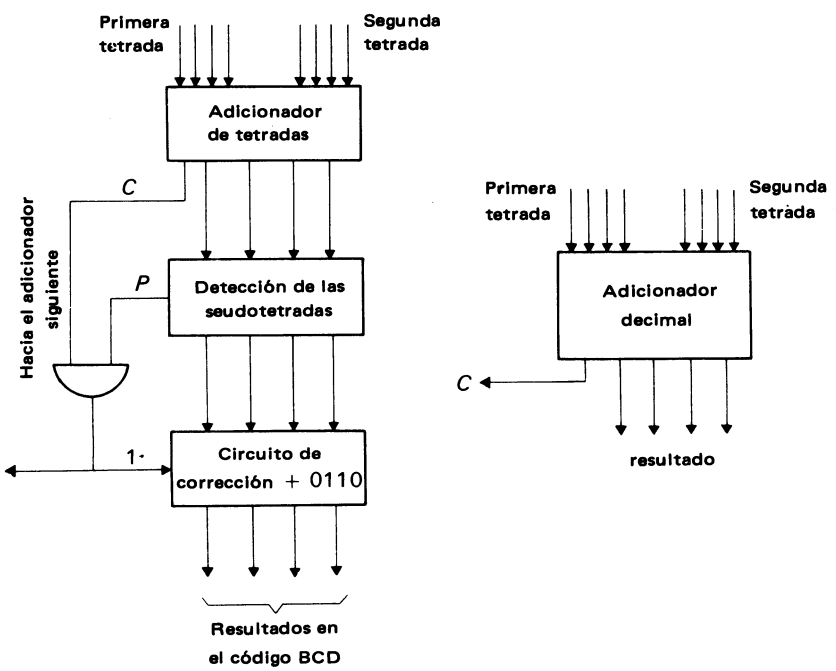


Fig. 32. *Esquema de principio y símbolo de un adicionador decimal*

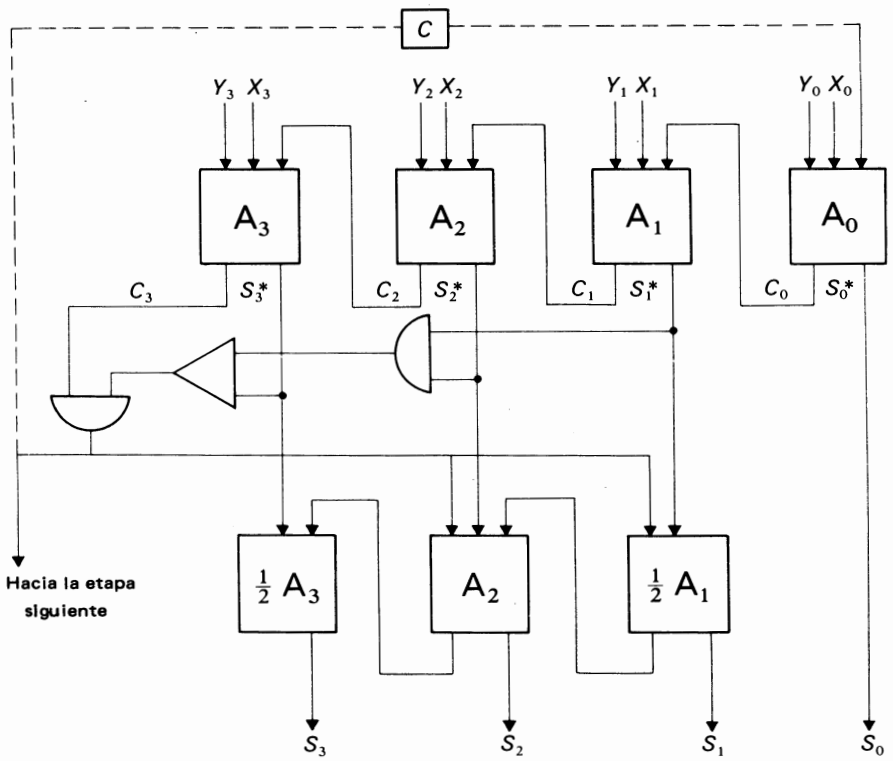


Fig. 33. *Adicionador en el sistema decimal (BCD)*
El circuito punteado corresponde al caso del adicionador decimal serie-paralelo

3.1 Aplicación de los diagramas de Karnaugh a cinco variables: determinación del circuito corrector de un adicionador decimal completo por medio del álgebra de Boole.

Igualmente podemos determinar directamente un adicionador completo por medio de un circuito transcodificador para convertir el código binario puro al código *BCD*. La figura 33 bis representa el cuadro de certeza de este transcodificador. *ABCD* representa el equivalente binario de la suma de los dos números, *E* el arrastre binario, *VWXY* el equivalente decimal codificado binario y *Z* el arrastre decimal. Estamos pues en presencia de cinco funciones de cinco variables que serán simplificadas por medio de los diagramas de Karnaugh (Fig. 33ter). La simplificación de los diagramas, da:

$$V = A\overline{BC} + CE\,,$$
$$W = \overline{A}B + BC + \overline{C}E\,,$$

Binario						Decimal codificado binario				
D	*A*	*B*	*C*	*D*	*E*	*V*	*W*	*X*	*Y*	*Z*
0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	0	1	0
2	0	0	1	0	0	0	0	1	0	0
3	0	0	1	1	0	0	0	1	1	0
4	0	1	0	0	0	0	1	0	0	0
5	0	1	0	1	0	0	1	0	1	0
6	0	1	1	0	0	0	1	1	0	0
7	0	1	1	1	0	0	1	1	1	0
8	1	0	0	0	0	1	0	0	0	0
9	1	0	0	1	0	1	0	0	1	0
10	1	0	1	0	0	0	0	0	0	1
11	1	0	1	1	0	0	0	0	1	1
12	1	1	0	0	0	0	0	1	0	1
13	1	1	0	1	0	0	0	1	1	1
14	1	1	1	0	0	0	1	0	0	1
15	1	1	1	1	0	0	1	0	1	1
16	0	0	0	0	1	0	1	1	0	1
17	0	0	0	1	1	0	1	1	1	1
18	0	0	1	0	1	1	0	0	0	1
19	0	0	1	1	1	1	0	0	1	1
20	0	1	0	0	1					
21	0	1	0	1	1					
22	0	1	1	0	1					
23	0	1	1	1	1					
24	1	0	0	0	1					
25	1	0	0	1	1					
26	1	0	1	0	1		∅			
27	1	0	1	1	1					
28	1	1	0	0	1					
29	1	1	0	1	1					
30	1	1	1	0	1					
31	1	1	1	1	1					

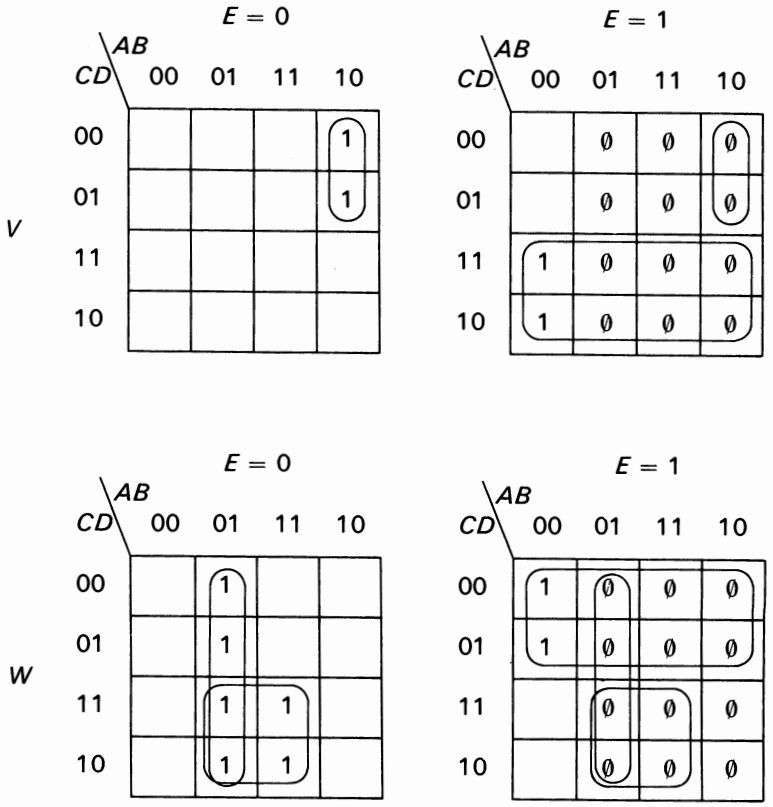
Fig. 33bis. *Cuadro de certeza del adicionador completo. E es el arrastre binario, Z el arrastre decimal*

$$X = \overline{C}E + AB\overline{C} + \overline{A}C\overline{E},$$

$$Y = D,$$

$$Z = E + AB + AC.$$

Obsérvese que los dos códigos son idénticos para los valores decimales comprendidos entre 0 y 9, y diferentes para los valores comprendidos entre 10 y 19 inclusive. Los valores de 20 a 31 representan las condiciones prohibidas. Basta con disponer a la salida del adicionador de las tetradas, que dan A,B,C,D y E, un circuito corrector definido por las ecuaciones lógicas precedentes. Suministrando éste la suma en decimal codificado binario. Obsérvese, no obstante, que este circuito es menos económico que el circuito corrector estudiado anteriormente. Mientras que aquél solamente intervenía cuando estaba en presencia de una seudotetrada o cuando existía un arrastre más allá del resultado, el transcodificador tiene en cuenta todas las combinaciones, incluso aquellas que no dan lugar a una corrección.



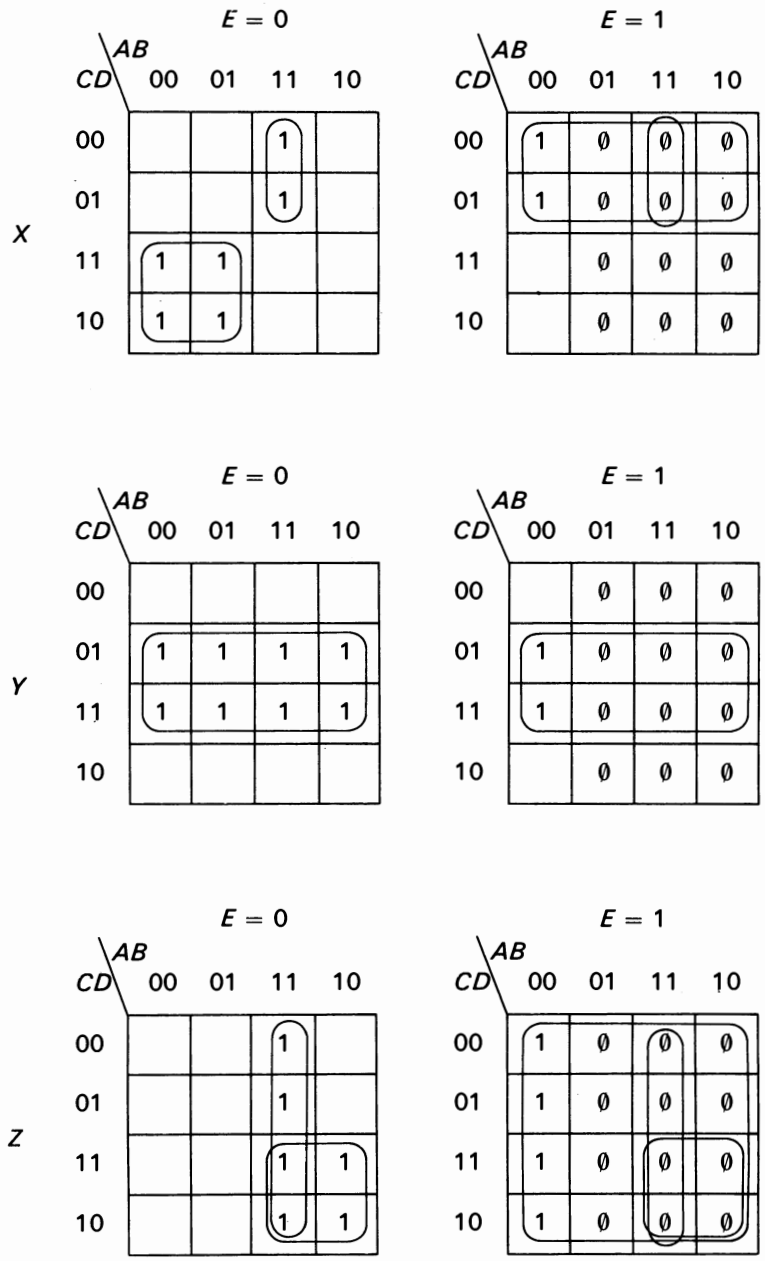


Fig. 33ter. *Diagramas de Karnaugh del adicionador completo*

3.2 Adicionador decimal que utiliza el código Aiken

La configuración de un adicionador cambia según el código utilizado para representar los dígitos decimales. Como ejercicio podemos tratar el caso de un adicionador decimal que utiliza el código Aiken o código 2*, 4, 2, 1 (fig. 34). Obsérvese que este código es ponderado y complementario. A partir del valor decimal 5 el peso 2* es prioritario sobre el peso 2. Las seudotetradas relativas a este código están representadas en la figura 35. Examinando atentamente este código, se observa que cuando la suma de dos tetradas coinciden con una

Valor decimal	2 S_3^*	4 S_2^*	2 S_1^*	1 S_0^*
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	1	0	1	1
6	1	1	0	0
7	1	1	0	1
8	1	1	1	0
9	1	1	1	1

Fig. 34. *Código Aiken*

2 S_3^*	4 S_2^*	2 S_1^*	1 S_0^*
0	1	0	1
0	1	1	0
0	1	1	1
1	0	0	0
1	0	0	1
1	0	1	0

Fig. 35. *Seudotetradas del código Aiken*

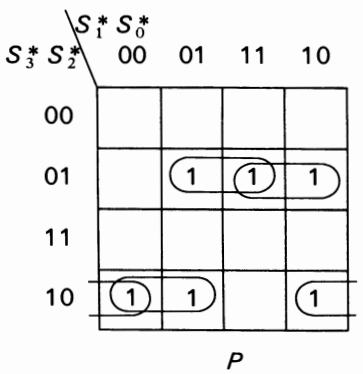


Fig. 36. *Diagrama de Karnaugh del circuito de detección de las seudotetradas en el adicionador decimal que utiliza el código Aiken*

seudotetrada y que no existe arrastre más allá del orden más elevado, basta con añadir $6_{10} = 0110$ a la suma para obtener el resultado exacto.

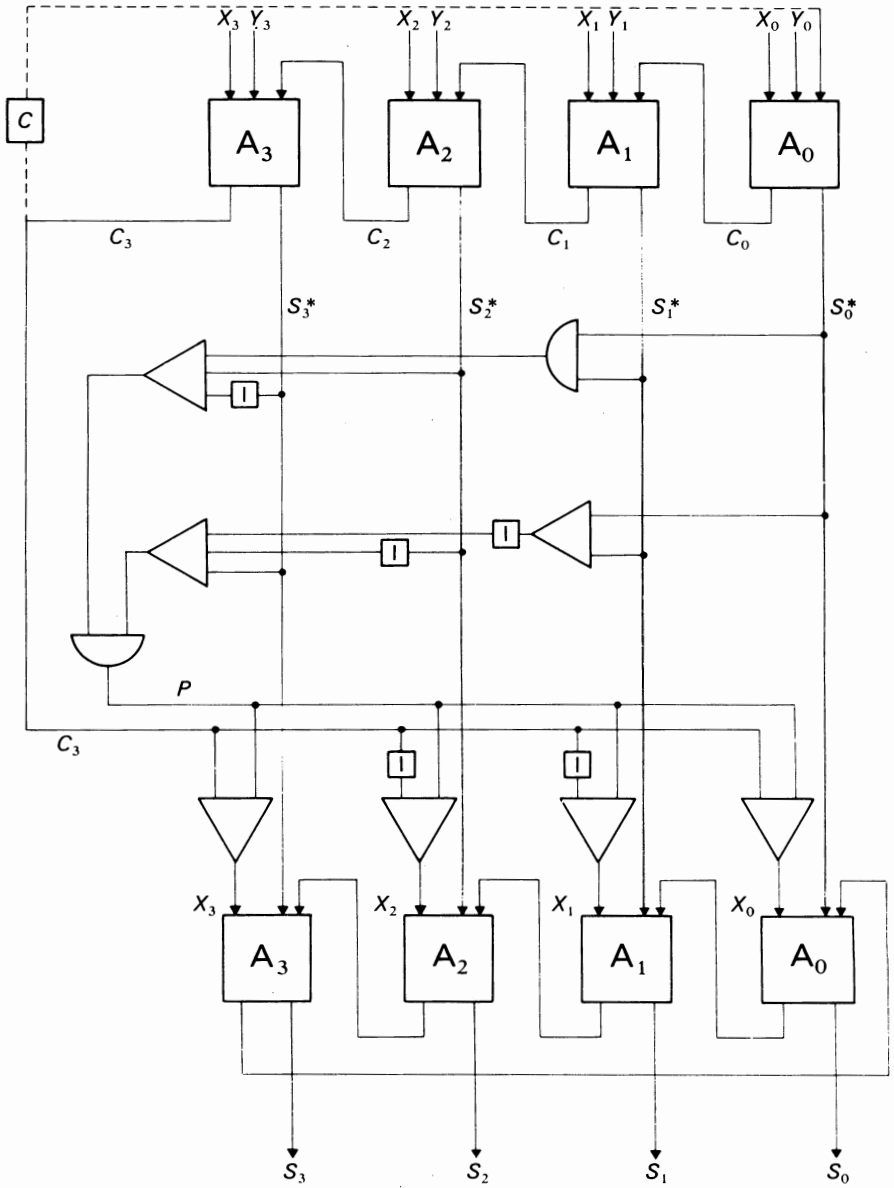


Fig. 38. *Adicionador serie-paralelo que funciona según el código Aiken*

Si existe un arrastre más allá del orden más elevado del resultado, estando en presencia de una seudotetrada, bastará con quitar $6_{10} = 0110$ de la suma para obtener el resultado. Llamemos P a la salida del circuito detector de seudotetradas. Según la figura 35, obtendremos el diagrama de Karnaugh de la figura 36.

La lectura de este diagrama da:

$$P = S_2^* \, \overline{S}_3^* (S_0^* + S_1^*) + S_3^* \, \overline{S}_2^* \overline{(S_0^* \, S_1^*)} \, .$$

Ahora será necesario realizar un circuito que añada o quite $6_{10} = 0110$ según los valores de C y de P, se trata del arrastre más allá del orden más elevado. Llamemos $X_3 \, X_2 \, X_1 \, X_0$, a la salida de cada uno de los circuitos que efectúan esta función sobre cada uno de los pesos S_i^* según el *cuadro de certeza de la figura* 37. La sustracción tiene lugar cuando $P = 1$ y $C_3 = 1$ por adición del complemento de $6_{10} = 0110$. El esquema del adicionador está representado en la figura 38.

El circuito punteado solamente se utiliza en el caso de un adicionador en serie-paralelo.

P	C_3	X_3	X_2	X_1	X_0
0	0	0	0	0	0
0	1	0	0	0	0
1	0	0	1	1	0
1	1	1	0	0	1

Fig. 37. *Cuadro de certeza del circuito corrector*

4. Los comparadores

Se entiende por comparador, un dispositivo capaz de comparar dos números binarios, determinando si éstos son iguales, o de lo contrario cuál de ellos es el más grande y cuál el más pequeño. Existen dos tipos de comparadores: en paralelo y en serie. En el primer caso se trata de comparar dos números que están fijos en dos registros, en el segundo caso, la información contenida en los dos registros es desplazada y ambos números son comparados peso por peso.

4.1 Comparadores en paralelo

Consideremos dos números binarios $A_1 \, A_2 \, A_3 \ldots A_i$ y $B_1 \, B_2 \, B_3 \ldots B_i$, en los que A_i y B_i son los dígitos de orden i de los dos números. Ambos números serán iguales si son idénticos todos los pesos entre ellos. Realizándose entonces la

coincidencia de todos los pesos: $X_1 = A_1 \odot B_1$

$$X_2 = A_2 \odot B_2$$
$$X_3 = A_3 \odot B_3$$
$$\vdots$$
$$X_n = A_n \odot B_n .$$

Los dos números serán iguales si se verifican todas las condiciones simultáneamente. Llamando Z a la salida del comparador, tendremos:

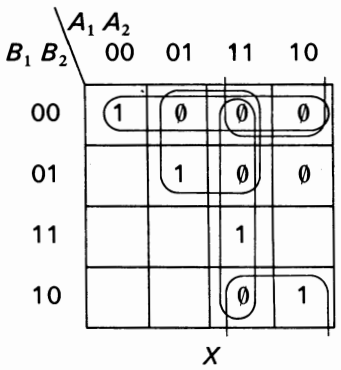
$$Z = X_1 X_2 X_3 \ldots X_n = (A_1 \odot B_1)(A_2 \odot B_2) \ldots (A_n \odot B_n) .$$

$Z = 1$ si los dos números son idénticos, $Z = 0$ si son diferentes. Otra manera de proceder es introduciendo el número A en un contador e incrementar éste hasta que $A = B$, estando B dispuesto en un registro. Obsérvese que en este caso A *nunca podrá ser superior a* B. Podemos pues imaginar un circuito lógico tal que la salida X responda a las condiciones siguientes:

$X = 0$ si $A < B$, $\quad X = 1$ si $A = B$, $\quad X = \emptyset$ si $A > B$.

Tomemos el caso de dos números, A y B, limitados a dos pesos. El cuadro de certeza es el siguiente:

A_1	A_2	B_1	B_2	X
0	0	0	0	1
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	$\emptyset$
0	1	0	1	1
0	1	1	0	0
0	1	1	1	0
1	0	0	0	$\emptyset$
1	0	0	1	$\emptyset$
1	0	1	0	1
1	0	1	1	0
1	1	0	0	$\emptyset$
1	1	0	1	$\emptyset$
1	1	1	0	$\emptyset$
1	1	1	1	1



Llevando este cuadro al diagrama de Karnaugh obtendremos las ecuaciones lógicas siguientes:

$$X = \overline{B}_1\,\overline{B}_2 + A_1\,A_2 + A_2\,\overline{B}_1 + A_1\,\overline{B}_2\,,$$
$$X = (A_1 + \overline{B}_1)\,(A_2 + \overline{B}_2)\,.$$

Siendo entonces posible la generalización para números de orden *n:*

$$X = (A_1 + \overline{B}_1)\,(A_2 + \overline{B}_2)\,(A_3 + \overline{B}_3)\,\ldots\,(A_n + \overline{B}_n)\,.$$

Sabiendo que A no será nunca superior a B, se tiene así un medio de verificar si $A < B$ o si $A = B$.

Consideremos ahora el caso más general en el que no se hace ninguna hipótesis sobre A y B. Deseamos saber si $A = B$, $A < B$ o $A > B$. Se comparan sucesivamente todos los pesos A_i y B_i de los dos números comenzando por los pesos de mayor orden, sirviéndose de un circuito de coincidencia. Si los dos números son idénticos todos los pesos son sucesivamente examinados. Si los números son diferentes, existe una posición en la que los dos dígitos de ambos números no coinciden. No siendo necesario continuar la comparación de los pesos más pequeños, pero sí asegurar cuál de los dos dígitos es 1. Si $B_i = 1$ y $A_i = 0$, entonces $B > A$; si $A_i = 1$ y $B_i = 0$, entonces $A > B$. El esquema del comparador está representado en la figura 39. Obsérvese que esta solución conduce a circuitos complejos si los números a comparar son muy grandes.

4.2 Comparación por sustracción

Otra solución consiste en sustraer uno de los dos operandos y examinar la existencia eventual de un arrastre más allá del peso de mayor orden del módulo del resultado. De este modo si $B > A$ y si se efectúa la diferencia $A - B$ estaremos **siempre en presencia de un arrastre C más allá del módulo del resultado.**

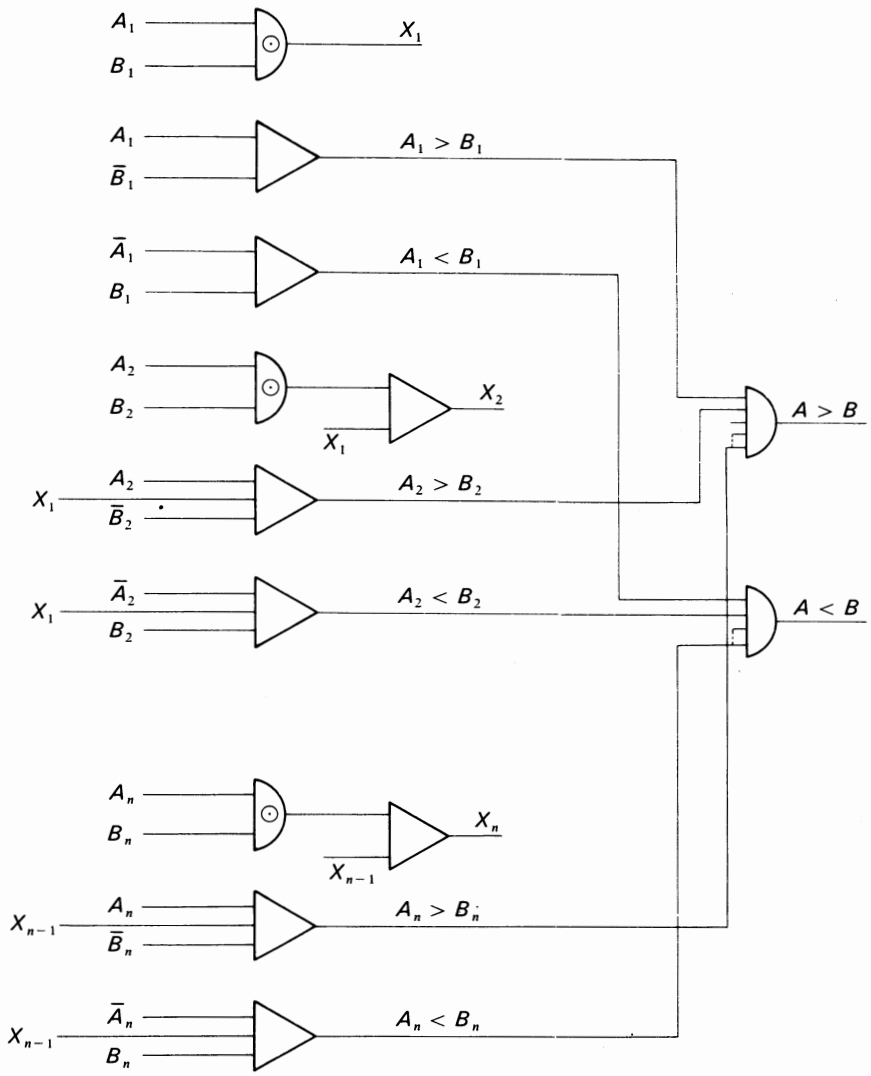
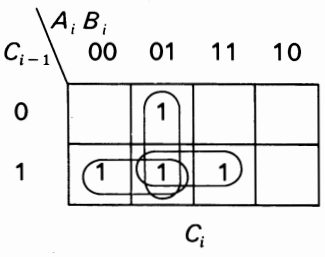


Fig. 39. *Esquema de un comparador en paralelo*

Si $B < A$, no habrá arrastre más allá del peso de mayor rango. La operación consiste en efectuar la diferencia, *sin preocuparse del resultado*, pero teniendo en cuenta los arrastres en cada etapa. El cuadro de certeza y el diagrama de Karnaugh, son los siguientes:

A_i	B_i	C_{i-1}	C_i
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1



La lectura del diagrama, da:

$$C_i = \overline{A}_i B_i + \overline{A}_i C_{i-1} + B_i C_{i-1} ,$$

que se transmitirá al orden superior. El esquema general del comparador está representado en la figura 40.

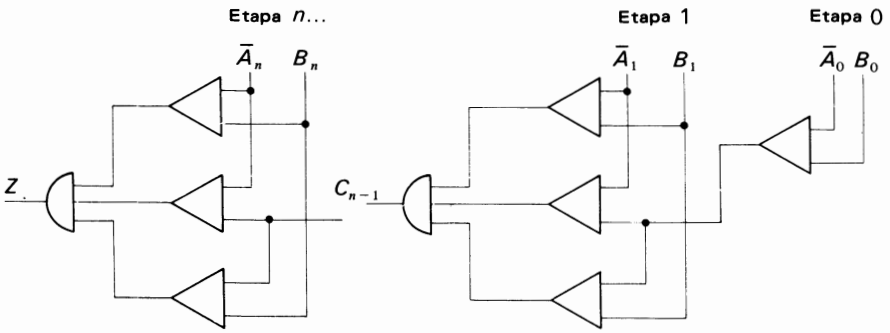


Fig. 40. *Comparador en paralelo por sustracción de B*
Si $Z = 1, B > A$; *si* $Z = 0, A = B$ *o* $A > B$.

4.3 Comparadores en serie

En este caso la comparación se efectúa solamente sobre un peso, obteniendo la comparación de los demás por desplazamientos de los dos números en los registros de desplazamientos. Si los números a comparar tienen cada uno n posiciones binarias se necesitarán como máximo n tiempos elementales para efectuar la comparación.

Si únicamente se desea saber si los dos números son idénticos o diferentes utilizaremos el esquema de la figura 41. La báscula B se pone en 1 al principio de la operación. Un impulso de avance se aplica a los dos registros así como a la entrada de la báscula. Si dos de los pesos de los dos números son diferentes, la báscula es puesta a cero indicando que los números son diferentes. Se suprime el impulso de avance después de un número de desplazamientos igual al número de posiciones idénticas de cada uno de los dos números. Si se desea obtener informaciones complementarias se utilizará el esquema de la figura 42, que representa un comparador por sustracciones sucesivas. Se utiliza entonces un generador de arrastres idéntico al estudiado para el comparador en paralelo. Una báscula C memoriza un arrastre eventual. Obsérvese la conexión cruzada entre los dos registros A y B. Un impulso de avance es aplicado sobre los dos registros y la báscula C (Fig. 42).

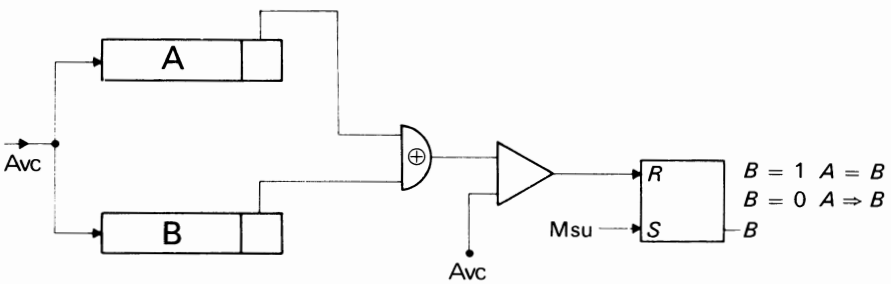


Fig. 41. *Comparador serie. $B = 1$, $A = B$, $B = 0$, $A \neq B$*

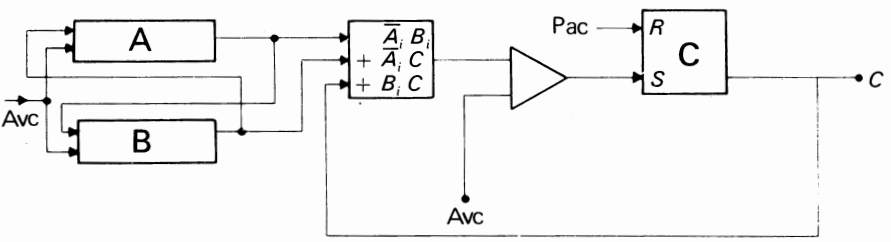


Fig. 42. *Comparador por sustracciones sucesivas*

Al principio de la operación la báscula C es puesta a cero, a continuación se hace $A - B$ efectuando los desplazamientos necesarios. Si después de n desplazamientos $C = 1, B > A$. Si $C = 0$, existen dos posibilidades $A = B$ o $A > B$. Obsérvese que en este momento B está en A y viceversa, no obstante si se efectúa nuevamente n desplazamientos se realiza $B - A$. Si C permanece en cero, tenemos $A = B$; si $C = 1$, tenemos $A > B$.

Un tercer método similar al estudiado para los comparadores en paralelo consiste en comparar sucesivamente dos a dos todas las posiciones de los dos números comenzando por los pesos de mayor orden. El esquema está representado en la figura 43. El desplazamiento se efectúa de derecha a izquierda. Un circuito "O exclusivo" colocado a la salida de los registros señala dos dígitos diferentes. Una báscula B se pone en 1 al principio de la operación. Si después de n desplazamientos $B = 1$, los dos números son iguales. Por el contrario, si en el orden i los dos dígitos A_i y B_i son diferentes, la báscula B es puesta a cero y suprimidos los impulsos de avance. Seguidamente se examinan las dos salidas, $\overline{A}B$ o $A\overline{B}$. Según que una o la otra estén en 1 tendremos $A < B$ o $A > B$.

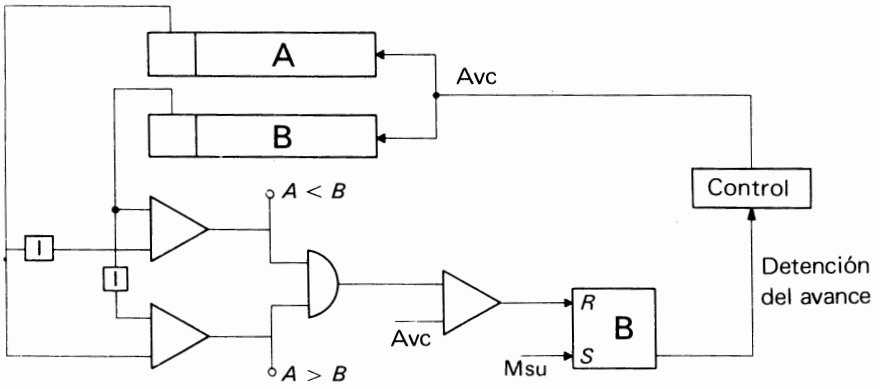


Fig. 43. *Comparación de dos números por comparaciones sucesivas de las diferentes posiciones de ambos números*

5. Los transcodificadores en serie

Hemos estudiado al principio del capítulo los transcodificadores en paralelo directamente por circuitos lógicos elaborados por medio del álgebra de Boole o por una decodificación previa en el sistema decimal seguida de una codificación en el nuevo código.

Igualmente se puede obtener un cambio de código en serie utilizando los registros de desplazamientos. Este método es mucho más económico en el caso en que los números codificados tengan un número importante de posiciones, aunque resulta más lento.

5.1 Conversión del código binario reflejado al código binario puro

Hemos estudiado en el capítulo 1 el método manual que permite pasar del código binario reflejado al código binario puro. Recuérdese que basta con efectuar la suma sin arrastre ("O exclusivo") de los sucesivos dígitos del número expresado

en el código binario reflejado empezando por los pesos de mayor orden. La primera suma sin arrastre se efectúa con el peso de mayor orden y el dígito 0 situado a la izquierda del número. La figura 44 representa el esquema de principio de este tipo de transcodificador, el número a transcodificar está en un registro de desplazamientos. La báscula *B* se pone a cero al principio de la operación. Simulando de este modo el 0 que es colocado a la izquierda del número. Seguidamente acumula, al ritmo de los impulsos de avance, las sumas sucesivas sin arrastre que son introducidas, por una parte en el adicionador sin arrastre ("O exclusivo"), por otra parte en el registro de desplazamientos comenzando por el peso de menor orden. Si el número tiene n posiciones, se necesitarán $n + 1$ impulsos de avance para obtener el número en el registro de desplazamientos en binario puro.

5.2 Conversión del código BCD (decimal) al código binario puro por multiplicación por 10

Se trata esta vez de transformar un número expresado en el código decimal *BCD*, en su equivalente en binario puro. Recuérdese que cada dígito representa un símbolo decimal comprendido entre 0 y 9. Luego será representado por una tetrada de cuatro dígitos binarios. Un número en el código *BCD* está constituido por una serie de tetradas a las que se atribuyen los pesos $10^0, 10^1, 10^2 \ldots 10^n$. En estas condiciones, llamando T_i las tetradas de orden i podemos expresar un

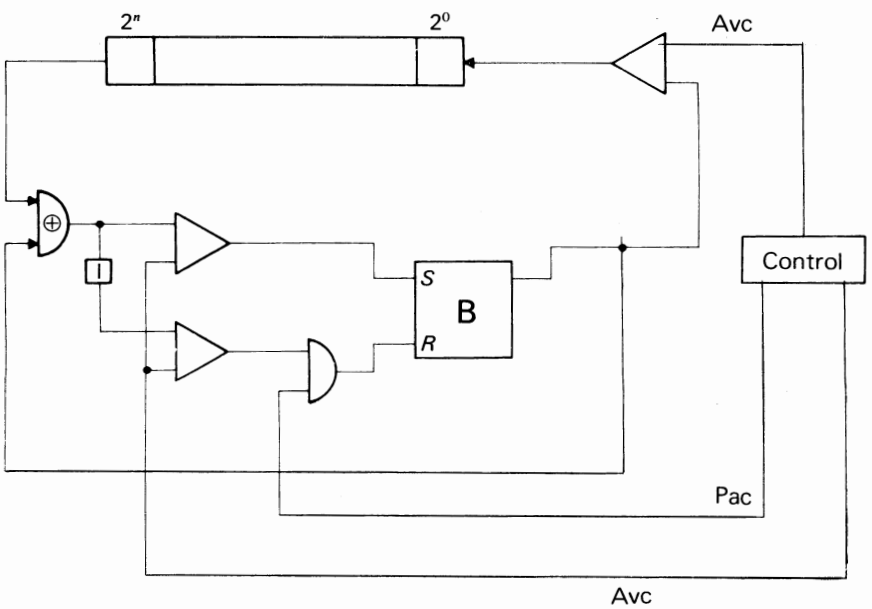


Fig. 44. *Paso del código binario reflejado al código binario puro*

número N bajo la forma

$$N = T_5\,10^5 + T_4\,10^4 + T_3\,10^3 + T_2\,10^2 + T_1\,10^1 + T_0\,,$$

que podremos escribir en la forma siguiente:

$$N = ((((10\,T_5 + T_4)\,10 + T_3)\,10 + T_2)\,10 + T_1)\,10 + T_0\,.$$

De este modo podemos calcular N que será expresado en binario puro. El algoritmo es el siguiente: multiplicar por 10 la tetrada del peso de mayor orden y añadir la tetrada siguiente continuando de este modo hasta que sea añadida la última tetrada T_0. El resultado representa el equivalente binario del número decimal expresado en el código *BCD*. Más adelante estudiaremos los diferentes modos de mecanizar los algoritmos de la multiplicación estudiados en el tomo I. En el presente caso la multiplicación por una constante (10) es muy fácil de realizar. En efecto, la multiplicación por 10 de un número binario se obtiene adicionando este número a sí mismo habiéndolo desplazado previamente de dos posiciones a la izquierda y añadiendo un 0 a la derecha del resultado.

Ejemplo. *Multiplicar* 1101 *por* 10 *decimal.*
Según lo explicado anteriormente, obtenemos:

$$\begin{array}{r} 1\,1\,0\,1 \\ +\ 1\,1\,0\,1 \\ \hline 1\,0\,0\,0\,0\,0\,1 \\ +\ 0 \\ \hline 1\,0\,0\,0\,0\,0\,1\,0 \end{array}$$

que es el equivalente binario de 130.

El esquema del transcodificador está representado en la figura 45. Contiene un registro de desplazamientos de n posiciones destinado a recibir el número en el código binario puro, un registro B de tres posiciones para realizar el desplazamiento de dos posiciones a la izquierda del número registrado en A antes de realizar la suma, un registro C para recibir sucesivamente en paralelo las tetradas del número expresado en el código *BCD*, comenzando por los pesos de mayor orden, dos adicionadores, uno necesario para la multiplicación por 10, el otro para la adición de la tetrada siguiente, un circuito de control que suministra los impulsos de avance necesarios para los desplazamientos de los números. No se han representado en el esquema las básculas de arrastre de cada adicionador. Al principio de la operación el registro A es puesto a cero, la tetrada de mayor orden es introducida en el registro C y el sistema de control suministra n impulsos de avance. La tetrada se encuentra entonces en la parte derecha del registro A $(0 \times T_{n+1} + T_n)$. La segunda tetrada del número decimal es introducida, bajo un mando del circuito de control, en el registro C. El conjunto de los registros recibe

entonces n impulsos de avance. El registro A contiene en este momento $10\,T_5 + T_4$ si T_5 es la tetrada del peso de mayor orden del número decimal. De este modo se continúa hasta el momento en que la tetrada del peso de menor orden es introducida en C. Un último desplazamiento de n posiciones del conjunto de los registros hace aparecer el resultado final en el registro A que contiene entonces el equivalente en código binario puro, del número decimal expresado en el código *BCD*. Obsérvese que el resultado es obtenido después de $(n + 1)\,T$ tiempos elementales, donde T es el número de tetradas.

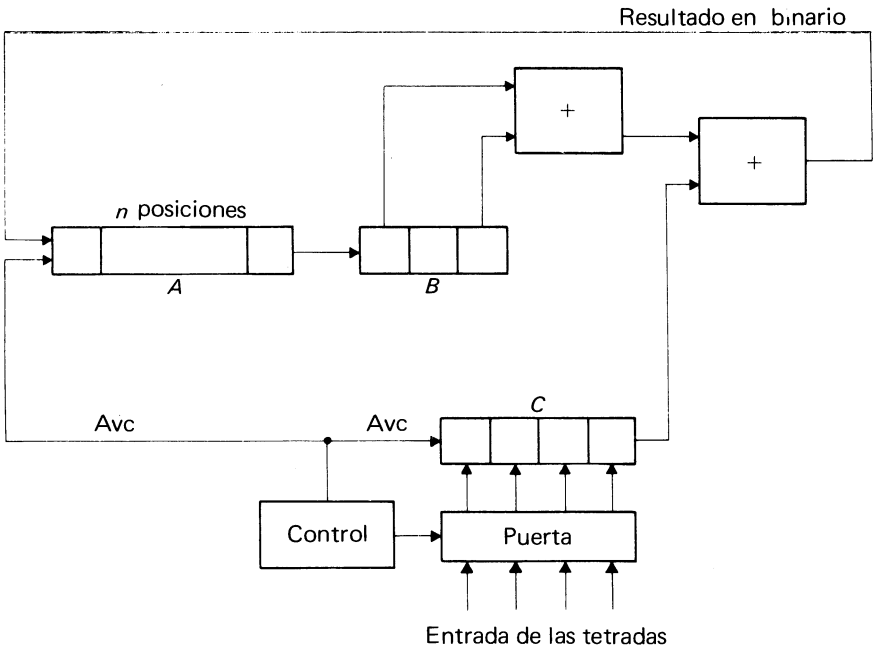


Fig. 45. *Convertidor, Código BCD ⇒ código binario puro*

Ejemplo. *Transcodificar, al código binario puro, el número decimal* 125 *expresado en código BCD:*

0 0 0 1 0 0 1 0 0 1 0 1
1 2 5

La primera tetrada de mayor orden 0001 es introducida en el registro C, el registro A es puesto a 0. Después de n impulsos de avance la tetrada 0001 está cuadrada a la derecha en el registro A. La segunda tetrada 0010 es introducida en el registro. Los cálculos se desarrollan entonces del modo siguiente:

	0 0 0 1
	0 0 0 1
	0 0 0 1 0 1
	+ 0
1 × 10	0 0 0 1 0 1 0
	+ 0 0 1 0
(1 × 10) + 2	0 0 0 1 1 0 0
	0 1 1 0 0
	0 1 1 1 1 0 0
	+ 0
((1 × 10) + 2) 10	0 1 1 1 1 0 0 0
((1 × 10) + 2) 10 + 5	+ 0 1 0 1
	0 1 1 1 1 1 0 1

Capítulo 3

La unidad de memoria

1. Introducción

Nos proponemos estudiar en este capítulo la memoria interna del calculador. Se entiende por memoria interna, un dispositivo que conserva durante todo el tiempo del tratamiento, las instrucciones y los datos necesarios para el desarrollo del programa. Se trata pues de un dispositivo que debe almacenar un gran número de informaciones. Hemos estudiado ya en el tomo I un cierto número de dispositivos que tienen la propiedad de conservar en memoria una información. Como por ejemplo los diferentes registros de la máquina, de tal modo que se podría imaginar la memoria interna del calculador como una asociación de una gran cantidad de estos registros, estando cada uno de ellos localizado por una dirección. Naturalmente es necesario poder acceder, según las necesidades, a cada uno de estos registros separadamente. El desarrollo de los circuitos integrados permite prever actualmente una solución de este tipo. No obstante, la mayor parte de los ordenadores actualmente en servicio poseen memorias de núcleos de ferrita.

1.1 Memoria de núcleos magnéticos

El elemento fundamental de una memoria magnética es el núcleo magnético. Se trata de un elemento muy pequeño (2 mm de diámetro aproximadamente) de material ferromagnético (fig. 1). Cada núcleo es susceptible de conservar en memoria un dígito binario. En una memoria habrá pues tantos núcleos magnéticos como dígitos binarios a almacenar.

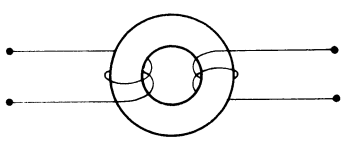


Fig. 1. *Núcleo magnético*

Se dispone sobre este núcleo un arrollamiento primario que será recorrido por una corriente eléctrica de sentido variable. El núcleo es entonces el soporte de un flujo magnético cuyo sentido dependerá del sentido de la corriente. Las propiedades del núcleo magnético dependen de su *curva de histéresis.* Por curva de histéresis se entiende, la representación de la inducción B en el núcleo en función de la intensidad del campo magnético H creado por la corriente en el arrollamiento primario (figura 2). Una corriente positiva en el arrollamiento primario da lugar a un campo magnético $+ H$ positivo en el núcleo. Inversamente una corriente negativa en el arrollamiento provocará un campo magnético negativo $- H$. Haciendo crecer positiva o negativamente la corriente en el arrollamiento se hace crecer simultáneamente el campo y la induc-

ción magnética. No obstante, este fenómeno no es lineal observándose una saturación de la inducción para valores elevados del campo magnético. Llamando $-B_M$ a la inducción máxima correspondiente al campo máximo negativo $-H_M$. Esta inducción máxima corresponde a la saturación del núcleo, ya que un nuevo aumento del campo no se traduce por un nuevo aumento de B. Así observamos que si el campo magnético varía de $-H_M + H_M$ recorremos la parte inferior de la curva. Por el contrario, si pasamos de $+H_M$ a $-H_M$ será la parte superior de la curva de histéresis la que nos interesa. Si se aplica al núcleo una corriente suficiente para provocar su saturación ($+B_M$ o $-B_M$) éste volverá a la posición $+B_R$ o $-B_R$ siguiendo el sentido de la corriente. B_R representa la inducción remanente del núcleo. Distingamos dos estados del núcleo: $+B_R$ y $-B_R$.

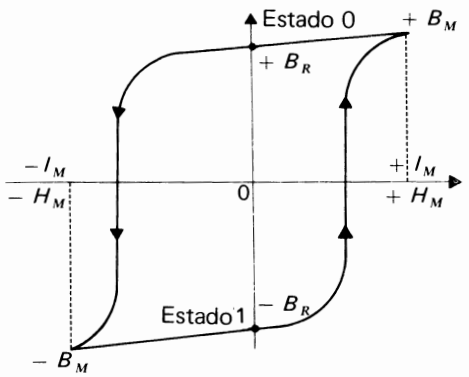


Fig. 2. *Curva de histéresis de un núcleo magnético*

Atribuiremos el valor lógico 1 a $-B_R$ y el valor lógico 0 a $+B_R$. De este modo para escribir un 1 en el núcleo magnético bastará con aplicar una corriente primaria $-I_M$ correspondiente al campo magnético $-H_M$. Entonces el núcleo magnético conservará el valor 1 indefinidamente, después de la desaparición de la corriente. Existe otro arrollamiento dispuesto sobre el núcleo destinado a leer su contenido, es decir a determinar si su estado es 0 ó 1. Consideremos el caso de un núcleo en estado 1 ($-B_R$). Aplicando una corriente $+I_M$ correspondiente a $+H_M$, el punto de funcionamiento del núcleo se desplaza sobre su ciclo de histéresis de $-B_R$ a $+B_M$. Si el núcleo está en estado 0, esta misma corriente provoca un desplazamiento del punto de funcionamiento de $+B_R$ a $+B_M$. En el primer caso la variación del flujo es grande, mientras que en el segundo caso esta variación es muy pequeña. Esta variación de flujo induce en el segundo arrollamiento, llamada línea sensible, una fuerza de electromotriz proporcional a la variación del flujo magnético (ley de Lenz).

En el primer caso obtenemos una f.e.m. inducida grande en la línea sensible que nos indica la presencia de un 1 en el núcleo. En el segundo caso la f.e.m. inducida es muy pequeña: el núcleo contiene un 0. De esta forma disponemos de

un modo de leer el contenido del núcleo. *Obsérvese, no obstante, que la información contenida en el núcleo desaparece después de la lectura.* En efecto, *un núcleo en estado* 1, después de la lectura de su contenido, pasa al estado 0. Pero la información en la memoria generalmente debe ser permanente. Luego será necesario restituir al núcleo su contenido después de la lectura. Para ello se procede del modo siguiente. La f.e.m. inducida en la línea sensible, provocada por el paso del núcleo del valor 1 al valor 0, será detectada y amplificada, utilizándose después para poner en 1 una báscula *R-S* o *T*. Transfiriéndose de este modo el estado del núcleo a la báscula. Es posible, entonces, por una segunda transferencia, volver a escribir un 1 en el núcleo, que de esta forma encuentra nuevamente su estado anterior. Esta operación será estudiada más adelante.

1.2 Selección de un núcleo

Acabamos de examinar cómo un núcleo es susceptible de poseer uno de los dos estados 0 ó 1, y cómo se efectúa la lectura de su contenido. Pero en la realidad, una memoria contiene gran número de núcleos que pueden llegar a más del millón. Es pues necesario poder seleccionar entre estos núcleos aquellos que nos interesan. Estos están generalmente agrupados por "palabras". Se entiende por palabra al máximo número de dígitos que pueden ser simultáneamente tratados por la máquina. Estas palabras están dispuestas generalmente, en la memoria, en forma de datos y de instrucciones. *Es también necesario poder acceder a cada una de estas palabras separadamente.* Lo cual se efectúa por medio de los circuitos de selección, que permiten elegir una de estas palabras entre las demás de la memoria. Antes de estudiar cómo se efectúa la selección de una palabra, examinemos cómo se obtiene la selección de un núcleo entre un grupo de núcleos. Generalmente, una memoria de núcleos está constituida por una asociación de *matrices*, comprendiendo cada una un número idéntico de núcleos. La interconexión de estas matrices es tal que eligiendo un núcleo de una matriz se elige también todos los núcleos que ocupan la misma posición relativa en todas las matrices. Se selecciona

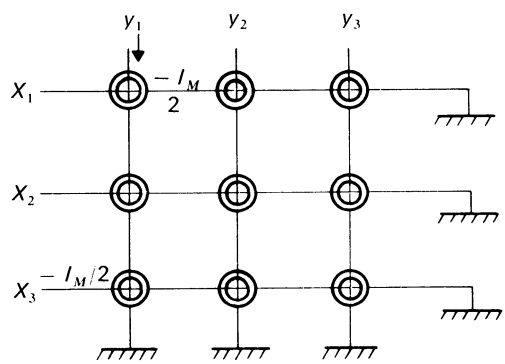


Fig. 3. *Matriz de núcleos en un plano*

así una palabra de memoria que comprende tantos dígitos binarios como matrices haya en la memoria. Examinemos primeramente la selección de un núcleo en un plano formando matriz (fig. 3). Supongamos que todos los núcleos de la matriz están en el estado 0, y deseamos escribir un 1 en el núcleo $x_2\ y_1$. Obsérvese que todos los núcleos en la matriz están referenciados por la intersección de una línea $x\ y$ de una columna y. Comuniquemos a la línea x_2 y a la columna y_1 una corriente $-I_M/2$. El núcleo $x_2\ y_1$ en la intersección de los dos hilos será recibido por una corriente $-I_M$. Por lo tanto pasará del estado 0 a 1. Los demás núcleos $x_1\ y_1, x_3\ y_3, x_2\ y_2, x_2\ y_3$ son recorridos solamente por una corriente $-I_M/2$ que no es suficiente para hacer pasar los núcleos interesados al estado 1. Permaneciendo pues a 0. *De esta forma disponemos de un modo de seleccionar un núcleo entre los demás de un mismo plano.* Comunicando una corriente $-I_M/2$ a una línea x_i y a una columna y_i seleccionamos el núcleo que está en la intersección de la línea x_i de la columna y_i. Si este núcleo estaba anteriormente en 1 se conservará su estado 1. Por el contrario, si su estado era 0, pasará al estado 1. Inversamente aplicando una corriente $I_M/2$ a la línea x_i y a la columna y_i se selecciona siempre el núcleo $x_i\ y_i$. No obstante si está en estado 1, pasará al estado 0, pero no cambiará de estado si anteriormente estaba en 0. Un núcleo que cambia de estado es recorrido por una variación de flujo bastante grande, que se traduce por una f.e.m. inducida si se dispone en cada núcleo una línea sensible.

Leer un núcleo, consiste en hacer pasar el núcleo considerado del estado 1 *al estado* 0.

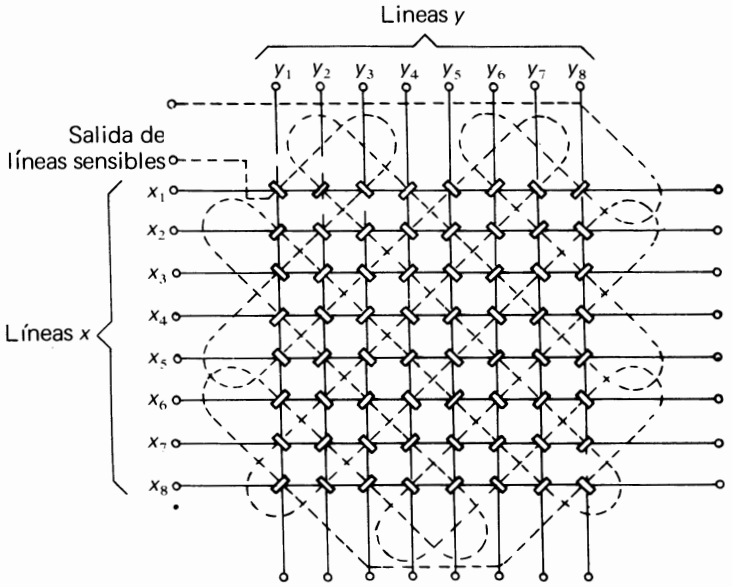


Fig. 4. *Plano de memoria*

De este modo suministra sobre la línea sensible una f.e.m. inducida. Los núcleos que no cambian de estado, es decir, los que permanecen en 0, suministran una f.e.m. inducida muy pequeña. Detectando esta f.e.m. se dispone de un medio para leer el contenido de un núcleo. Obsérvese, como hemos dicho antes, que la lectura de un núcleo destruye su contenido. Será pues necesario prever después de la lectura un procedimiento que permita reconstituir el contenido de cada núcleo. En una memoria la línea sensible pasa por todos los núcleos de cada plano. *Luego tendremos tantas líneas sensibles como planos o matrices haya en la memoria.* La f.e.m. inducida es de un valor muy pequeño (800 mV), por lo que será necesario prever amplificadores para poder explotar los resultados de la lectura. El esquema de la figura 4 da un ejemplo de una matriz de núcleos que comprende 8 líneas y 8 columnas, es decir, un total de 64 núcleos. Estos están alineados según ejes diferentes de un núcleo al otro con el fin de obtener una mejor compensación de las tensiones parásitas producidas por los núcleos que no cambian de estado en el curso de la lectura. Estos núcleos sufren una pequeña variación de flujo magnético suministrando sobre la línea sensible tensiones inducidas parásitas que son compensadas por un particular alineamiento de los núcleos.

1.3 Memoria tridimensional

El tipo de memoria que hemos estudiado consiste en un plano de n líneas por n columnas permitiendo seleccionar un núcleo entre los n^2 núcleos presentes. De tal modo que, como hemos visto, es necesario leer una palabra de varios dígitos binarios que comprende varios núcleos. Para obtener este resultado se asocian varios planos uniendo entre ellos en serie las diferentes líneas x y las diferentes columnas y de tal modo que un impulso de corriente aplicado por ejemplo a la línea x_1 del primer plano atraviesa las líneas x_1 de todos los planos sucesivos. Igualmente un impulso de corriente aplicado a la columna y_2 del primer plano atravesará las columnas y_2 de todos los planos.

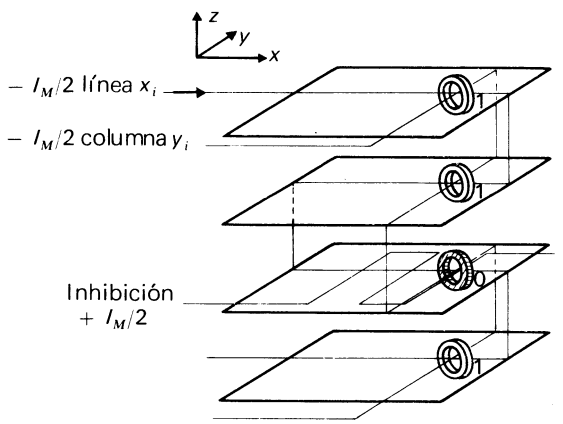


Fig. 5. *Asociación de matrices para realizar una memoria de núcleos*

Utilizando una matriz de 8 x 8 = 64 núcleos, se asociarán 10 matrices entre ellas del modo descrito para leer o escribir palabras de 10 bits en la memoria. En este caso cada impulso de corriente atravesará 80 núcleos. Cada matriz comprende una línea sensible separada. Estas no están conectadas entre ellas por lo que habrán tantas líneas sensibles como matrices en la memoria. Obsérvese igualmente que el número de líneas sensibles es igual al número de bits de cada palabra. La figura 5 representa una configuración de este tipo. *De este modo se seleccionan todos los núcleos que tienen la misma posición relativa en cada uno de los planos.* Un problema se presenta cuando se desea escribir un 0 en un núcleo que forma parte de una palabra. Según la dirección de la palabra se seleccionará la línea x_i y la columna y_i. Todos los núcleos que ocupen la misma posición relativa $x_i y_i$ en cada uno de los planos se seleccionarán y si están sobre 0 pasarán a 1, ya que cada uno de ellos recibe una corriente.

$$\left(-\frac{I_M}{2}\right)+\left(-\frac{I_M}{2}\right)=-I_M\,.$$

Si se desea escribir un cero en uno de los núcleos correspondiente a un plano bien determinado, será necesario hacer pasar por este núcleo una corriente adicional $+I_M/2$, con el fin de transformar la corriente total que recorre este núcleo a

$$-I_M+\frac{I_M}{2}=-\frac{I_M}{2}\,.$$

En este caso la corriente que recorre este núcleo no es suficiente para escribir un "1". Se llega a este resultado asociando a cada plano una línea llamada *"línea de inhibición"* que atraviesa todos los núcleos de cada plano y que es recorrida por una corriente $+I_M/2$ cuando se desea escribir un cero en el núcleo de posición relativa $x_i\, y_i$ correspondiente a este plano. Esta línea de inhibición está generalmente cableada transversalmente en cada plano (fig. 5). Así escribiremos el número binario 1011 de abajo hacia arriba. El núcleo rayado contiene un 0 ya que solamente ha sido sensibilizada la línea de inhibición correspondiente a este plano. Resumiendo, cada núcleo está atravesado por cuatro hilos: dos hilos de selección, una línea sensible y una línea de inhibición. El número de planos de la memoria es igual al número de bits de cada palabra. *El producto del número de líneas x_i por el número de columnas y_i representa el número total de palabras susceptibles de ser almacenadas en memoria.*

1.4 Selección de una palabra en la memoria.

Para seleccionar una palabra en la memoria será necesario disponer de generadores de corriente para las líneas x_i e y_i. Estos generadores deberán suministrar una corriente *en los dos sentidos* $\pm I_M/2$ según que se desee leer o

escribir una palabra en la memoria. Deberá disponerse igualmente de tantos amplificadores de lectura y generadores de inhibición como planos y matrices haya en la memoria. Finalmente cada palabra leída deberá ser memorizada en un registro (registro de memoria) antes de ser nuevamente escrita en la memoria. Las posiciones del registro que estén en 0 sensibilizarán los generadores de inhibición correspondientes a estas posiciones. En la escritura se registrará pues un cero en los núcleos correspondientes a estos planos. La figura 6 representa un esquema muy general del sistema de direccionamiento y de selección de una palabra de n bits. La memoria comprende 64 palabras, o sea 8 líneas x (x_0 a x_7) y 8 columnas y (y_0 a y_7). La lectura de las palabras en la memoria se efectúa del modo siguiente. Se introduce en los registros dirección x e y la dirección de la palabra en la memoria, es decir, la posición relativa de cada núcleo de un plano que debe ser leído. Supongamos que la palabra que se desea leer corresponde a los núcleos que ocupan la posición relativa siguiente en cada uno de los planos: línea 2, columna 5. Introduciremos en forma binaria el valor 010 en el registro de dirección x y 101 en el registro de dirección y.

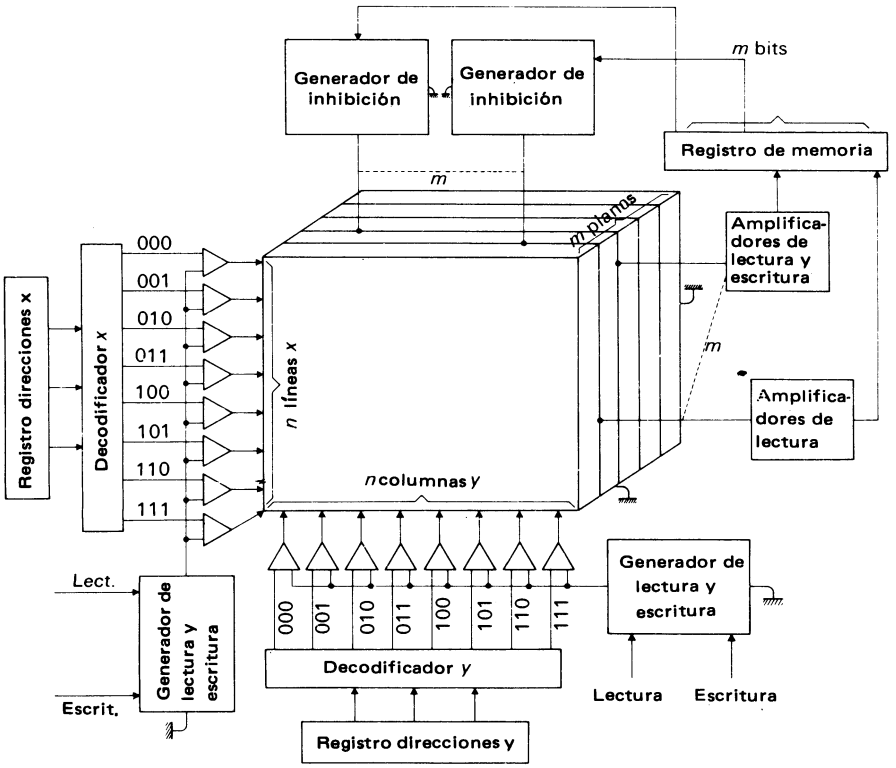


Fig. 6. *Esquema general de la organización de una memoria de núcleos de ferrita, de 8 líneas x 8 columnas, o sea 64 palabras de m bits*

El decodificador x sensibiliza la línea de salida 010 (fig. 6) y el decodificador y la línea de salida 101. Un mando de lectura es enviado seguidamente a los generadores de lectura x e y emitiendo cada uno una corriente $+I_M/2$. Los núcleos de cada plano que ocupan la posición relativa $x_2\ y_5$ emitirán un impulso de salida sobre la línea sensible si previamente contenían un "1". Recuérdese que solamente estos núcleos son los que cambian de estado en el momento de la lectura.

Estos emiten una tensión suficiente a los amplificadores de lectura. La tensión es amplificada para introducir el contenido del núcleo en la posición correspondiente del registro memoria. De este modo, después de la lectura, la palabra cuya dirección es $x_2\ y_5$ se encuentra en el registro memoria. Supongamos que la palabra en la memoria tenía la configuración siguiente 101001. La memoria comprende 6 matrices o planos diferentes de 8 x 8 = 64 núcleos. Lo que hace un total de 64 palabras de 6 bits. Después de la lectura la palabra 101001 se encuentra en el registro memoria que comprende 6 posiciones, es decir 6 básculas *R-S*. Se utiliza entonces la salida complementaria de las básculas que están en cero para sensibilizar los generadores de inhibición. Un mando de escritura es seguidamente enviado, inmediatamente después de la lectura o bien al final del ciclo de decodificación de la instrucción. Los generadores de escritura X e Y emiten un impulso de corriente $-I_M/2$ y los generadores de inhibición correspondiente a las posiciones 0 del registro memoria emiten un impulso de corriente $+I_M/2$. De este modo, escribimos un "1" en los núcleos que ocupan la misma posición relativa $x_2\ y_5$ en todos los planos, salvo en aquellos que son recorridos por una corriente de inhibición en los que escribimos un cero.

1.5 Realización del registro memoria

El registro memoria está constituido por tantas básculas como bits haya en una palabra de memoria. Cada báscula recibe en su entrada S un impulso procedente de los amplificadores de lectura. Así se transfiere una palabra al registro memoria en el momento de la lectura. Cada salida complementaria de las básculas está unida a un generador de inhibición, conectado éste al mismo tiempo con cada plano de la memoria. Como hemos visto, suministra una corriente $+I_M/2$; no obstante, cuando se produce un mando de escritura sobre los núcleos correspondientes a los planos para los cuales el contenido de las básculas del registro *RM* es nulo, serán recorridos por una corriente $-I_M/2$. Permaneciendo sobre 0.

Los núcleos correspondientes a los planos para los cuales el contenido de las básculas del registro es 1 se pondrán en 1. La palabra se escribe de este modo en la memoria de nuevo. El esquema de realización está representado en la figura 7. El mando de los amplificadores de lectura no se efectúa inmediatamente después de la aparición del impulso de mando de los generadores de lectura. En efecto, los núcleos no seleccionados son recorridos por una corriente $+I_M/2$ mientras que

los núcleos seleccionados serán recorridos por una orriente $+I_M$. Los núcleos no seleccionados emiten sobre la línea sensible una tensión inducida muy pequeña pero cuyos efectos se añaden. La tensión resultante puede ser de amplitud igual a la suministrada por el único núcleo seleccionado en un plano. Los planos en los que no se ha seleccionado ningún núcleo, pueden emitir un impulso que puede ser interpretado como un "1" por el amplificador de lectura. Para evitar este inconveniente se "activan" los amplificadores de lectura por medio de una "puerta". En efecto, las tensiones parásitas inducidas por los núcleos no seleccionados provienen de un fenómeno reversible (paso de $-B_M$ a B_R) y decrecen mucho más rápidamente que la tensión procedente de un núcleo que haya basculado completamente (paso de $+B_R$ a $-B_R$ por ejemplo) (Fig. 8). Basta, entonces, con sensibilizar los amplificadores de lectura después de la desaparición de las tensiones parásitas.

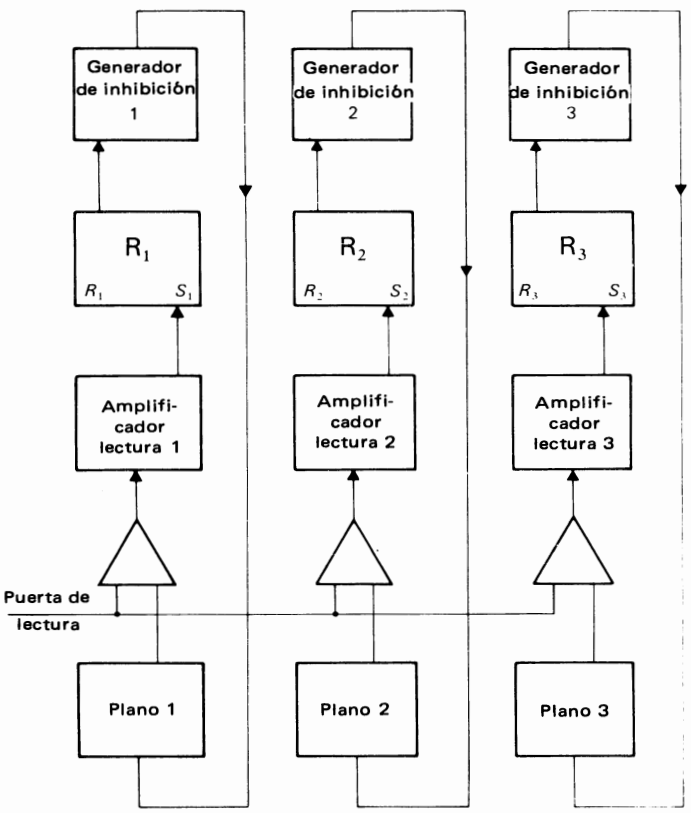


Fig. 7. *Registro de memoria y circuitos asociados*

1.6 Realización del registro y del decodificador de direcciones

El registro de direcciones recibe las sucesivas direcciones y los datos en la memoria que hay que decodificar para el desarrollo del programa. Contiene n básculas si 2^n es el número total de palabras en la memoria. Es seguido de un decodificador de direcciones que selecciona una línea y una columna en cada plano. El registro de direcciones está realizado por un registro único que contiene 2 partes distintas de $n/2$ básculas. En el caso de una memoria de $8 \times 8 = 64$ palabras tendremos en total 6 ($2^6 = 64$) básculas de las cuales tres estarán reservadas para el direccionamiento de las líneas x y las otras tres para el direccionamiento de las columnas y. La figura 9 da una representación de un registro de direcciones resultante de la yuxtaposición de dos registros x e y representado en la figura 6. Las direcciones sucesivas de las instrucciones proceden del contador de instrucciones (llamado también contador ordinal), las direcciones de los datos de la parte dirección de la instrucción.

El decodificador de direcciones está constituido por un decodificador clásico como el estudiado anteriormente en el capítulo 1. En la figura 10 reproducimos el esquema del decodificador de líneas y columnas del esquema de la figura 6.

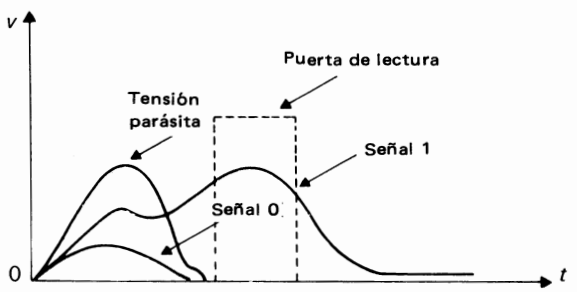


Fig. 8. *Representación por signos útiles y parásitos provocados por la línea sensible. En punteado la puerta de lectura de los amplificadores*

Dirección de líneas x	Dirección de columnas y
$n/2$	$n/2$

Fig. 9. *División del registro de direcciones en dos partes, para las líneas y las columnas*

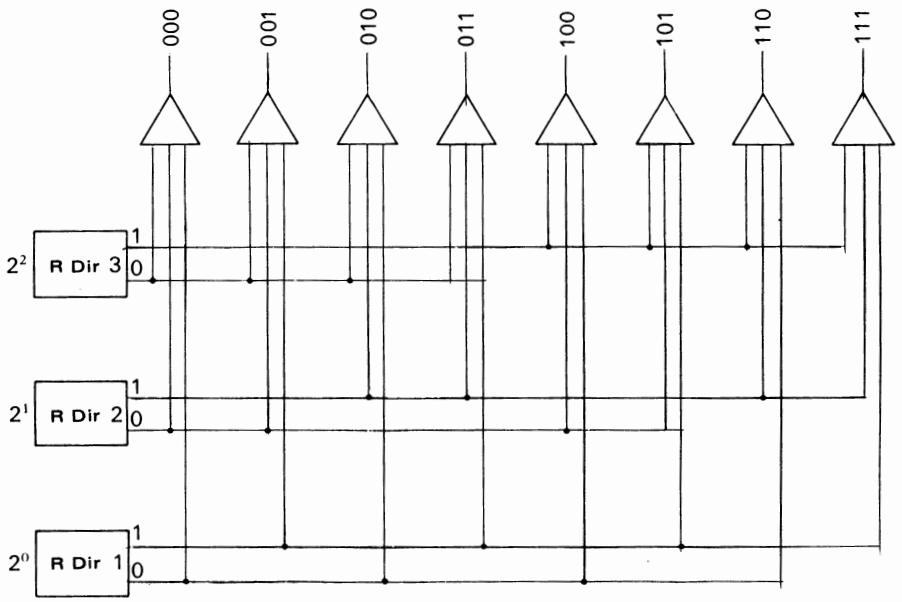


Fig. 10. *Decodificador de las líneas y columnas de la memoria de la figura 6*

1.7 Realización práctica del sistema de direccionamiento

El sistema de direccionamiento de la figura 6 es válido solamente para las memorias de pequeña capacidad. Se hace demasiado lento cuando la capacidad de la memoria aumenta. Consideremos por ejemplo una memoria de 10.000 palabras, comprendiendo 100 líneas x y 100 columnas y. Tenemos a la salida de cada decodificador x e y 100 circuitos "y" de potencia, controlados por un generador de escritura y de lectura de los cuales uno solo estará en funcionamiento. Podemos pues apreciar que este sistema no es económico. Para disminuir el número de circuitos "Y" a la salida de cada decodificador, se efectúa una codificación de los generadores y de los circuitos "Y". Estos se reemplazan prácticamente por interruptores de transistores. Las líneas y las columnas de cada plano se conectan siempre en serie, agrupándose por cuatro del lado del generador y de dos en dos del lado de los interruptores. Este método conduce a utilizar, en nuestro caso, dos generadores de lectura y de escritura. Obsérvese que los generadores G_1 x y G_1 y están conectados a las líneas y columnas 0, 1, 2, 3 y los generadores G_2x y G_2 y a las líneas y columnas 4, 5, 6, 7. En la salida del lado de los interruptores, las líneas y columnas 0,4-1, 5-2, 6-3,7 están conectadas dos a dos. De este modo se disminuye el número de circuitos "Y" a la salida de cada decodificador. Esto será particularmente sensible en el caso de memorias de gran capacidad. Observemos

igualmente que los generadores de lectura y de escritura intervienen en la codificación y que solamente sensibilizan dos grupos de cuatro líneas de cuatro columnas. Sólo trabajan durante la mitad del tiempo (Fig. 11).

1.8 Memorias de tres hilos

Las memorias que acabamos de estudiar son las llamadas de cuatro hilos. En efecto, cuatro hilos atraviesan cada núcleo: dos hilos destinados a la lectura y a la escritura, una línea sensible y una línea de inhibición.

Es posible reunir las dos últimas líneas en una sola que hará la función de línea sensible durante la lectura, y de línea de inhibición durante la escritura. En este caso bastará con prever una conmutación apropiada de la línea sobre el generador de inhibición en el momento de la escritura.

1.9 Memorias de selección lineal

El tipo de direccionamiento que acabamos de estudiar consiste en la selección de un núcleo que ocupa la misma posición relativa en todos los planos. Es un sistema de tres coordenadas: x, y, z (en la dirección de los planos). Otro método consiste en seleccionar una palabra en la memoria *por una sola línea,* mientras que en los ejemplos precedentes era necesario sensibilizar dos líneas x e y para seleccionar una palabra en la memoria. En este caso, si la memoria comprende 2n palabras de m bits tendremos un registro de direcciones de n pesos seguido de un decodificador de $2n$ salidas, de las cuales sólo una será sensibilizada. Los núcleos estarán colocados sobre cada una de estas líneas y atravesados por una corriente $\pm I_M$ para la grabación y la lectura de un 1. El esquema de principio de la organización de la memoria de este tipo está representado en la figura 12.

Cada núcleo es recorrido por una línea del registro de direcciones que selecciona una palabra entre las demás, por una línea de inhibición y una línea sensible. Hay tantos generadores de inhibición y amplificadores de lectura como bits en cada palabra de memoria. Por el contrario, sólo hay un generador de lectura y de escritura. La falsa salida de cada báscula del registro de memoria está conectada al generador de inhibición. El funcionamiento es el siguiente: en la lectura, el generador de lectura suministra un impulso de corriente $+I_M$ a la línea seleccionada por el decodificador de direcciones. Todos los núcleos de la palabra seleccionada son puestos a cero. Los que contenían un 1 suministran una fuerza electromotriz a la línea sensible, que amplificada por los amplificadores de lectura, pone en 1 la báscula del mismo peso del registro de memoria. Los núcleos que contenían un 0 no suministran ninguna tensión inducida de tal modo que las básculas correspondientes del registro de memoria permanecen en 0. En la escritura, el generador de escritura suministra una corriente $-I_M$ poniendo todos los núcleos en 1 salvo aquellos que corresponden a un 0 en el registro de memoria. En efecto, para estos pesos, el registro de memoria controla el suministro del generador de inhibición que proporciona entonces una corriente $+I_M/2$.

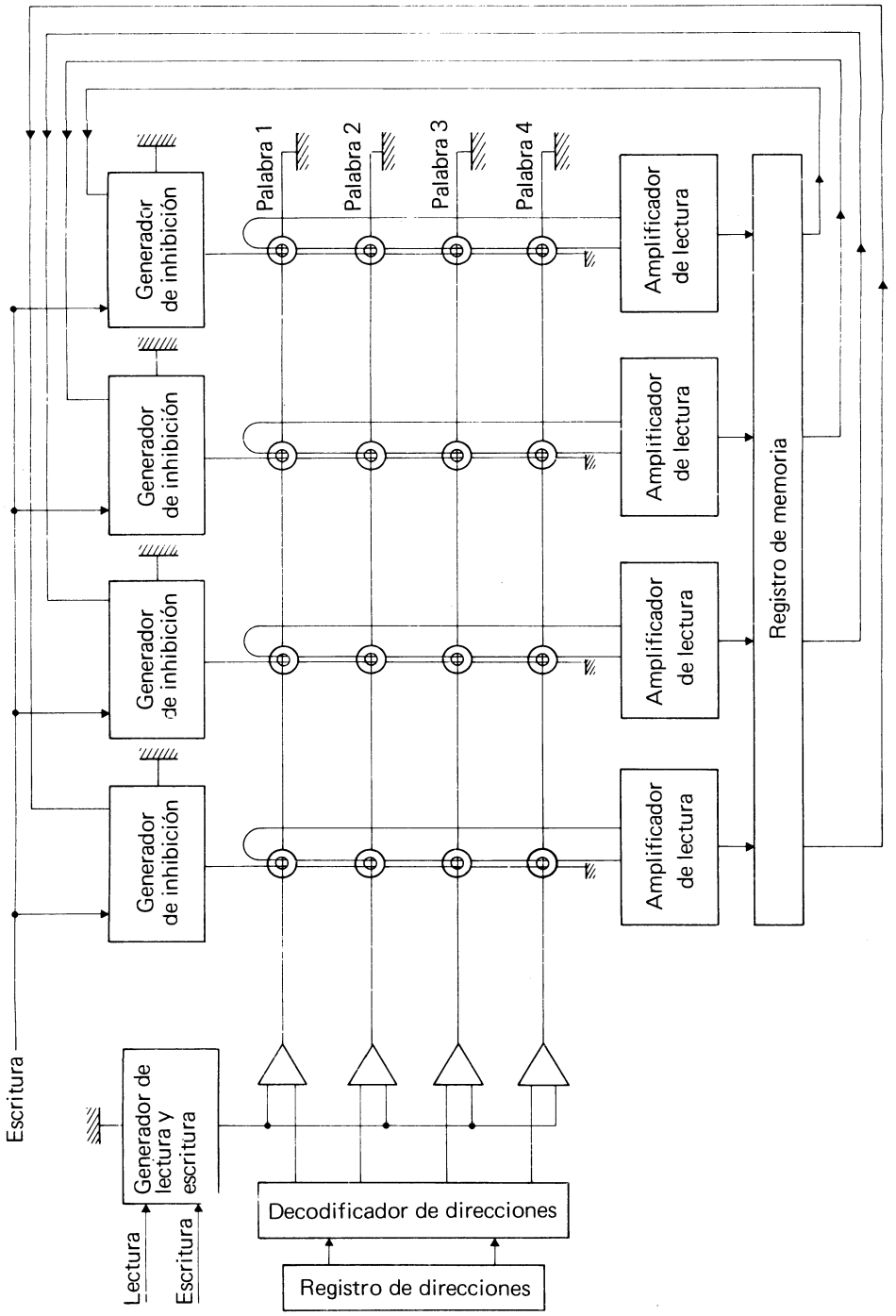


Fig. 12. *Memoria de selección lineal*

Los núcleos correspondientes son recorridos por una corriente resultante $-I_M/2$ permaneciendo en 0. La ventaja de una selección de este tipo reside en el hecho de que sólo los núcleos de la palabra seleccionada son recorridos por una corriente *en la lectura.* Por lo tanto no hay tensiones parásitas inducidas en la línea sensible. Luego podrán utilizarse corrientes de lectura y de escritura más elevadas con el fin de saturar aún más cada uno de los núcleos de la memoria. Se obtendrán de este modo tiempos de lectura y escritura más rápidos. El mayor inconveniente de este método reside en la complejidad del decodificador. Obsérvese, en efecto, que en este tipo de memoria la selección no se efectúa por coincidencia de dos corrientes. La selección debe efectuarse fuera de la memoria. Si la memoria contiene n palabras, será necesario disponer de un decodificador de n salidas mientras que en la memoria anterior cada decodificador poseía $\sqrt{n}$ salidas o sea $2\sqrt{n}$ salidas en total. Así en el caso de una memoria de $2^{10} = 1\,024$ palabras, el registro de direcciones contendrá 10 básculas y el decodificador (fig. 10), $10 \times 2^{10} = 10\,240$ diodos, lo que es una cantidad considerable. En el caso del sistema de direccionamiento por coincidencia, tenemos dos registros de direcciones (reunidos en uno solo) de cinco posiciones cada uno seguidos de un decodificador conteniendo $5 \times 32 = 160$ diodos o sea un total de 320 diodos en lugar de 10 240. Obsérvese la economía que se obtiene por el método de direccionamiento por coincidencia de dos corrientes. Es por esta razón que este último es el más utilizado. No obstante, en ciertos casos, el direccionamiento lineal es inevitable, por ejemplo en el caso de las memorias de películas delgadas o de memorias "muertas".

1.10 Memorias de dos hilos

Reuniendo nuevamente las líneas sensible y de inhibición en una sola línea común tendremos una memoria de dos hilos. Naturalmente, será necesario prever, como para la memoria de tres hilos, una conmutación de la línea común con el fin de pasar alternativamente sobre los amplificadores de lectura o sobre los generadores de inhibición.

1.11 Memorias muertas

Se entiende por memorias muertas un tipo de memorias en las que es imposible modificar la información registrada. Son memorias *"permanentes"* que conservan su contenido indefinidamente. La lectura de estas memorias no es destructiva y no existe ciclo de escritura.

1.11.1 *Memorias muertas de capacidad*

Entre las memorias muertas, que más se utilizan, tenemos las memorias de capacidad que permiten almacenar económicamente programas operacionales, programas de tests, tablas de funciones, y en general toda una parte de un programa que permanece inalterado durante el tratamiento (microprogramación).

La lectura de estas memorias se puede efectuar en fracciones de micro-segundos. La figura 11 representa una memoria de este tipo. Se han colocado en un circuito impreso las líneas de lectura. Estas líneas están conectadas separadamente a pequeños rectángulos metálicos que realizan la función de placas de condensador. Las líneas sensibles están realizadas del mismo modo pero dispuestas perpendicularmente a las líneas de lectura. De este modo se obtienen una serie de pequeños condensadores cuyas armaduras están conectadas a las líneas de lectura y a las líneas sensibles. Las armaduras están separadas por un dieléctrico generalmente de mylar. Si se desea escribir permanentemente un cero en la memoria, se suprimirá una de las armaduras; suprimiéndose al mismo tiempo la capacidad. La lectura se efectúa enviando un impulso sobre las líneas de lectura. En todos los lugares en los que exista una capacidad, el impulso de corriente será transmitido a la línea sensible: se detecta así un 1. Si no existe capacidad, la línea sensible no recibirá ninguna corriente: se detecta así un 0. Otro tipo de memoria muerta de capacidad

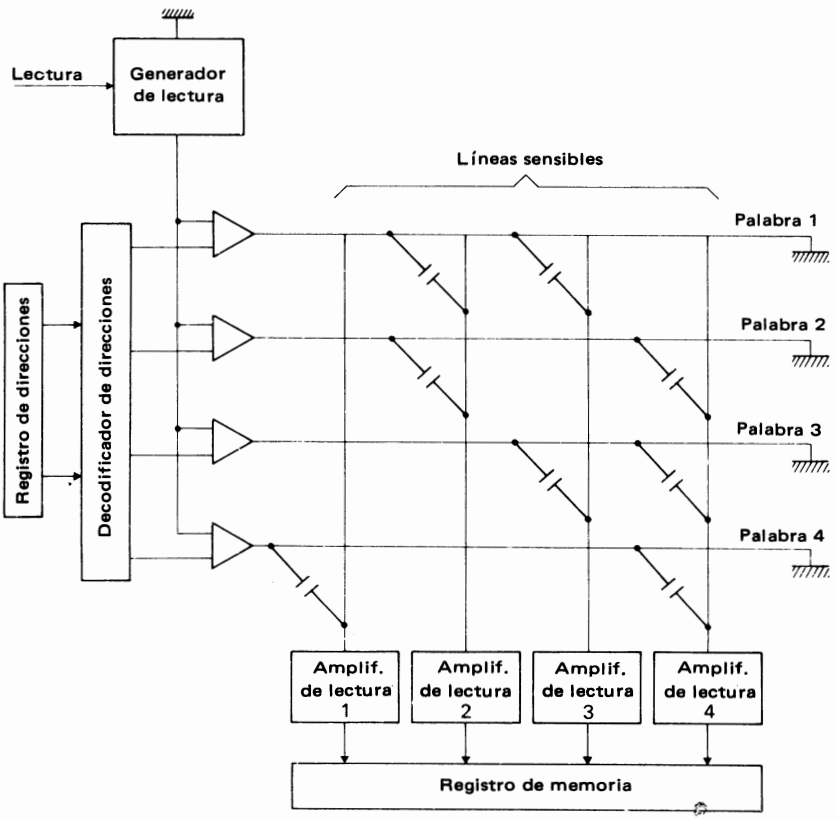


Fig. 13. *Selección y lectura de una palabra en una memoria muerta del tipo "tarjetas de capacidad"*

se obtiene introduciendo una placa metálica conectada a tierra y aislada por sus dos caras con una capa de mylar, entre los dos circuitos impresos que contienen las líneas de lectura y las líneas sensibles. De esta forma, se anula el efecto de capacidad entre las líneas de lectura y las líneas sensibles. Para introducir las palabras en este tipo de memoria, se perfora la placa metálica colocada entre las dos armaduras, sobre los lugares en los que se desee introducir un 1. La información es pues introducida de un modo *permanente* en este tipo de memoria perforando la placa metálica colocada entre las armaduras de un condensador conectadas a una línea de lectura y a una línea sensible.

El funcionamiento de una memoria muerta de capacidad es el siguiente: una matriz de decodificación selecciona la línea de lectura que constituye la palabra que debe ser leída. Cada línea sensible detectará un 0 ó un 1 según que exista o no una capacidad en la intersección de la línea de lectura y de la línea sensible (detección de los bits de cada palabra). Se trata pues de una selección de tipo lineal. La figura 13 representa una selección de este tipo. Se ha dispuesto una capacidad en los lugares en los que se ha introducido un 1 en la memoria. Así, la primera palabra contiene la información 0110, la segunda 0101, la tercera 0011, la cuarta 1001.

1.11.2 *Memorias muertas de transformadores*

Otro tipo de memoria permanente consiste en utilizar transformadores de núcleo magnético para cada bit de una palabra. Obsérvese no obstante, que el número de transformadores es igual al número de bits de una palabra sea cual sea el número de palabras en la memoria. La figura 41 representa el esquema de una memoria de este tipo. La selección de una palabra es *lineal.* El decodificador de direcciones tiene tantas salidas como palabras hay en la memoria. Cada salida atraviesa el transformador cuando se desea escribir un 1, evitando este transformador cuando se desea escribir un 0. La información es previamente introducida en los núcleos por el cableado de las líneas de lectura.

En el esquema de la figura 14, la palabra 1 es 1000, la palabra 2: 1101, la palabra 3: 0110 y la palabra 4: 0111.

1.12 Memorias de diodos túnel

Una de las características esenciales de una memoria es la rapidez con la que se puede leer o escribir una información. Esta característica impone además la velocidad máxima de tratamiento de las informaciones en la máquina. A fin de aumentar la rapidez de las memorias, se utilizan elementos que puedan cambiar de estado con gran rapidez: uno de estos elementos es el diodo túnel que puede cambiar de estado en algunos nanosegundos.

El diodo túnel es un elemento semiconductor que presenta una característica corriente-tensión semejante a la de la figura 15.

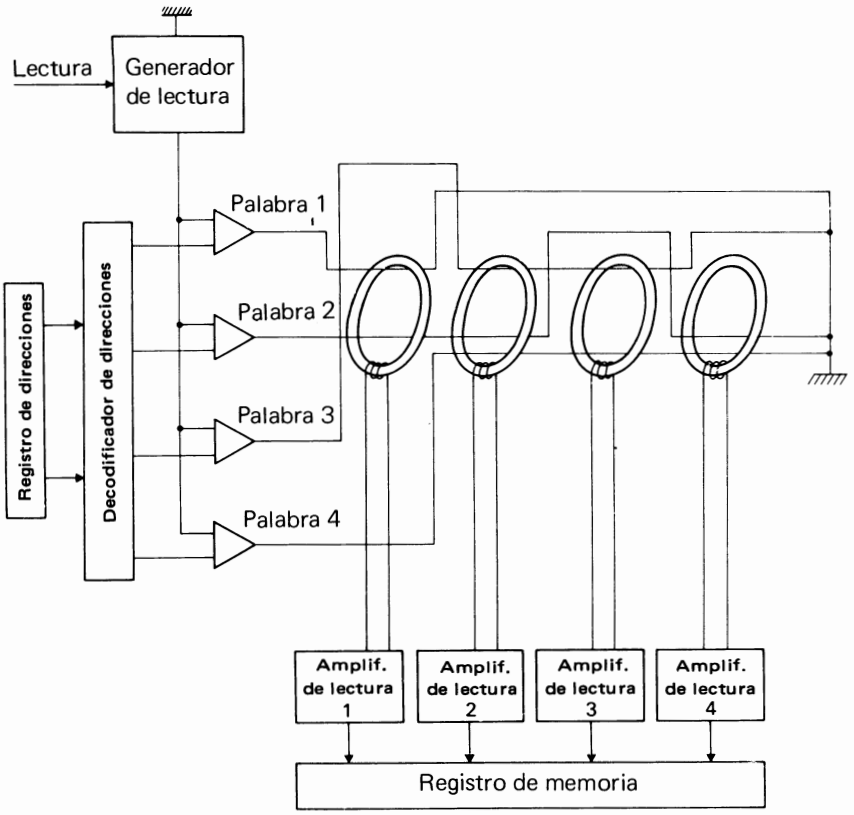


Fig. 14. *Selección y lectura de una palabra en una memoria muerta de transformadores*

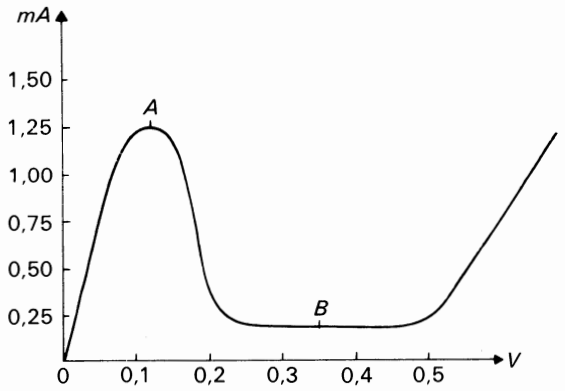


Fig. 15. *Característica corriente-tensión de un diodo túnel*

El símbolo de un diodo túnel está representado en la figura 16.

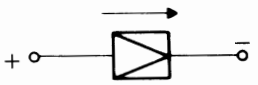


Fig. 16. *Símbolo de un diodo túnel*

Obsérvese la existencia de un máximo (a) en la curva así como un mínimo (B). Esto significa que entre A y B la corriente disminuye cuando la tensión aumenta: estamos en presencia de una zona de resistencia negativa. Antes del punto A y después del punto B, la corriente aumenta con la tensión aplicada: estamos en presencia de una zona de resistencia positiva. Conectemos ahora el diodo túnel a una resistencia R según el esquema de la figura 17.

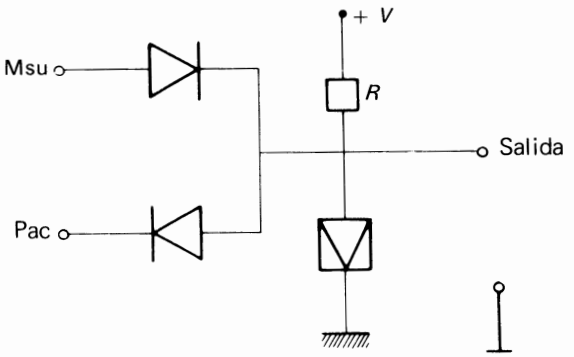


Fig. 17. *Circuito de mando de un diodo túnel*

La característica corriente-tensión es atravesada por la recta de carga R (Figura 17). Esta recta corta la característica en tres puntos A, B y C. El punto B situado en la zona de resistencia negativa es inestable y el diodo túnel no conservará este estado. Este tomará el estado A o C según el valor de la tensión aplicado en sus bornes. Atribuyamos al estado A el valor binario 0 y al estado C el valor binario 1. El diodo túnel basculará sucesivamente de A a C o de C a A, según el valor de la tensión V. Supongamos que el diodo túnel esté en el estado 0, es decir que su punto de funcionamiento esté situado en A. Si aplicamos una corriente suplementaria en el diodo por medio de una entrada M en la figura 16, la nueva recta de carga corresponderá a la recta punteada en la figura 18. El diodo túnel bascula rápidamente del punto A al punto D, suprimiendo luego la corriente suplementaria aplicada, el punto de funcionamiento pasa de D a C. El diodo túnel pasa del estado 0 al estado 1.

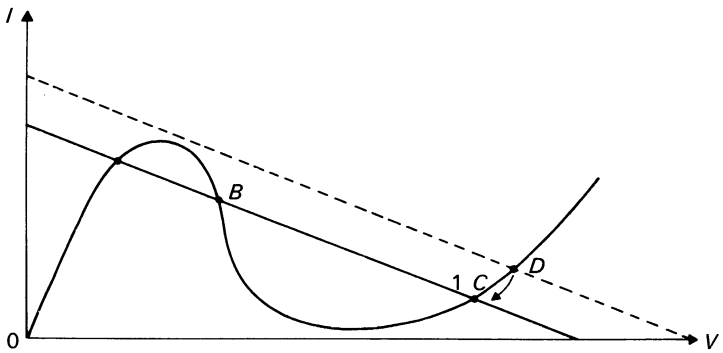


Fig. 18. *Paso del estado 0 al estado 1 de un diodo túnel por aumento de su débito*

Inversamente para volver al estado 0 en el punto *A* inyectaremos sobre la entrada Pac una corriente negativa opuesta a la que recorre el diodo. La nueva recta de carga corresponde a la recta punteada en la figura 19. El diodo túnel pasa totalmente del punto *C* al punto *E*. Si se suprime la inyección de corriente el punto de funcionamiento pasa de *E* a *A*, el diodo túnel pasa entonces del estado 1 al estado 0.

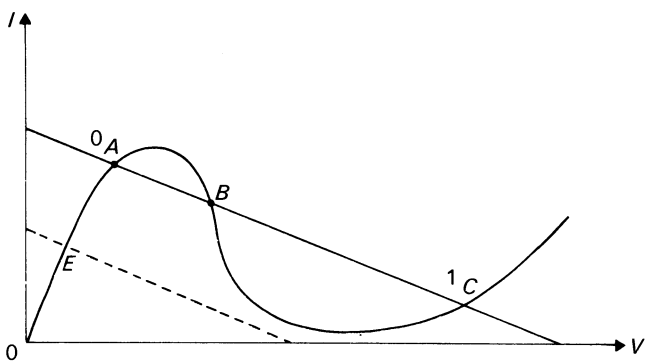


Fig. 19. *Paso del estado 1 al estado 0 de un diodo túnel*

El diodo túnel se comporta como una báscula *R-S*. Se utilizará la selección lineal de palabras en la memoria. La lectura se efectuará inyectando una corriente en la línea de selección cuyo sentido es tal que la tensión en los bornes del diodo disminuye a fin de que el punto de funcionamiento pase de 1 a 0. Solamente los diodos túnel cuyo estado era 1 serán sometidos a una variación importante de corriente que podrá ser amplificada por medio de un amplificador de lectura. La variación de la corriente será nula para los diodos previamente situados en

estado 0. Para escribir una palabra en la memoria se cambiará el sentido de la corriente en la línea de selección de tal modo que la tensión aumenta en los bornes de los diodos túnel para aproximarlos al máximo de la curva. Se inyectará a continuación una corriente suplementaria en los diodos que deban tomar en memoria un 1. Estos, y solamente éstos pasarán del estado 0 al estado 1.

Capítulo 4

La unidad aritmética y lógica

1. Introducción

Se llama unidad aritmética y lógica (U.A.L.) a la parte del ordenador que efectúa los cálculos necesarios al desarrollo del programa. Estos cálculos serán siempre reducidos a una serie de adiciones, de sustracciones, de multiplicaciones y de divisiones. Las operaciones más complicadas (extracciones de raíces, líneas trigonométricas, ecuaciones diferenciales) serán descompuestas de modo que puedan ser resueltas con las operaciones elementales efectuadas por la unidad aritmética. Esta preparación constituye el *"análisis numérico"* que tiene por objeto presentar el problema a resolver de un modo inteligible por la máquina. Seguidamente se escribe el programa que desarrolla en secuencia todas las operaciones elementales necesarias para la obtención del resultado. En estas condiciones la unidad aritmética, llamada también unidad central, debe ser capaz de resolver las cuatro operaciones elementales señaladas anteriormente. La constitución de la unidad aritmética es independiente de la naturaleza de las operaciones a efectuar. Solamente el número de registros utilizados y el funcionamiento de éstos difieren según la complejidad de las operaciones. Hemos establecido en el tomo I los algoritmos de las cuatro operaciones elementales. El lector puede referirse a ellos. Finalmente la unidad aritmética puede igualmente efectuar sumas y multiplicaciones lógicas.

2. Constitución de la unidad aritmética

La operación más compleja que realizará la unidad central será la división o eventualmente la extracción de la raíz cuadrada. Hemos visto en el capítulo 1 el algoritmo de la división. Es necesario prever tres registros: dos registros para el dividendo y el divisor, un registro para el cociente. La división se obtiene por una sucesión de adiciones y sustracciones: uno de los registros señalará los restos parciales de la división, éste será el acumulador.

De un modo general, la unidad aritmética (fig. 1) contendrá:

- un acumulador que registra uno de los números al principio de la operación así como todos los resultados intermedios;
- un registro multiplicador-cociente Q conectado al anterior para formar un único registro en el caso de la multiplicación y de la división;
- un registro memoria que conserva uno de los operandos durante toda la operación;
- un adicionador-sustractor que efectúa, bajo control del calculador, la adición o la sustracción del contenido del registro memoria R y del contenido del acumulador, el resultado aparece en el acumulador;

– un contador de desplazamientos C destinado a detener la operación en curso cuando ésta termine;
– un dispositivo destinado a señalar la invalidez de ciertas operaciones: división por 0, rebasamiento de capacidad del acumulador en el curso de las operaciones. Este será, en general, una báscula B Overflow. La unidad aritmética controla pues en cada instante la invalidez de las operaciones que ejecuta.

La interconexión de estos elementos varía según la naturaleza de la operación efectuada y el modo de representación de los números, tal como veremos en los párrafos siguientes. Supongamos que todos los números tienen una longitud fija de cinco dígitos comprendido el signo y que son siempre superiores a uno. El signo correspondiente pues al peso 2^4, los pesos 2^3 al 2^0 están reservados para el módulo del número.

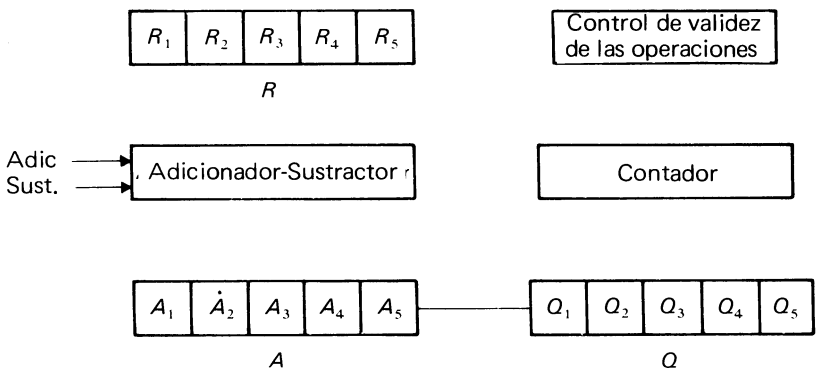


Fig. 1. *Constitución de la unidad aritmética*

3. Máquinas en paralelo y máquinas en serie

Generalmente se distinguen dos grandes categorías de máquinas: máquinas en serie y máquinas en paralelo.

En el primer caso los datos son tratados en serie, y en el segundo en paralelo. En las máquinas en paralelo, todos los dígitos de un número son tratados simultáneamente. En este caso es suficiente un tiempo elemental para efectuar una adición o una sustracción. Se trata de máquinas muy rápidas. Los dígitos de los dos números son adicionados simultáneamente teniendo en cuenta un arrastre eventual. El adicionador-sustractor contiene tantos adicionadores-sustractores elementales como dígitos haya que adicionar: se trata pues de un elemento relativamente complejo. Por el contrario, en las máquinas en serie, los dígitos de cada número son solamente accesibles sucesivamente y son adicionados unos

detrás de otros en un adicionador-sustractor único, éste es, pues, relativamente simple. No obstante, el resultado de la operación sólo se obtendrá al cabo de n tiempos elementales siendo n el número de dígitos de los dos números a adicionar: se trata de máquinas relativamente lentas. Observará el lector que la rapidez de ejecución de una operación se obtiene en detrimento de la complejidad de los circuitos. Examinaremos en este capítulo la realización de la unidad aritmética para los dos tipos de máquina.

4. Unidad aritmética y lógica en paralelo

4.1 Adición y sustracción

Supongamos que la unidad aritmética es del tipo en paralelo, lo que impone un circuito adicionador-sustractor así como un acumulador del tipo paralelo. El lector podrá referirse al capítulo 2 para estos problemas. La naturaleza de los circuitos varía, lo vimos anteriormente, según el modo de representación de los números. A continuación nos limitaremos al caso de los números representados en módulo y en signo o en suplemento a 2. El caso de los números negativos representados por sus complementos a 1 conduce a circuitos muy complejos y no se utiliza prácticamente.

4.1.1 *Números representados en valor absoluto y con signo*

En el caso de adiciones y sustracciones, el registro Q no se utiliza. Se supone que uno de los números está ya en el acumulador. La primera etapa consiste en introducir el número Y en el registro R. Las posiciones A_1 y R_1 son reservadas al

R_1	A_1	B Sgn $= A_1 \oplus R_1$	Operación efectuada	Operación sobre los módulos
0	0	0	Adición	Adición de los módulos
0	1	1		Sustracción de los módulos
1	0	1		Sustracción de los módulos
1	1	0		Adición de los módulos
0	0	0	Sustracción	Sustracción de los módulos
0	1	1		Adición de los módulos
1	0	1		Adición de los módulos
1	1	0		Sustracción de los módulos

Fig. 2. *Control de las operaciones de adición y sustracción*

signo de Y y de X. Un decodificador de las operaciones ordena "adición" o "sustracción" según la naturaleza del código de operación de la instrucción en curso. El desarrollo del algoritmo es el siguiente: primeramente se compara el signo de los dos números por medio de un circuito "O-exclusivo" que da $A_1 \oplus R_1 = 0$ si los dos signos son idénticos y $A_1 \oplus R_1 = 1$ si los signos son diferentes. Este resultado es memorizado en una báscula signo B Sgn. Según la naturaleza de la operación (adición o sustracción) impuesta por el decodificador de operaciones y el contenido de la báscula B Sgn, se efectuará una adición o una sustracción de los módulos según el cuadro de la figura 2.

La lectura de este cuadro pone de manifiesto que estamos en presencia de una adición de los módulos cuando los dos números tienen los mismos signos y que se realizará una adición, o que los signos de los dos números son diferentes y que se realizará una sustracción. Igualmente estaremos en presencia de una sustracción de los módulos cuando los dos números tienen signos diferentes y que se realizará una adición, o que los signos son idénticos y que se realizará una sustracción. Esto se expresa por las ecuaciones lógicas siguientes:

$$\text{Add mod} = \overline{\text{B Sgn}}\ \text{Add} + \text{B Sgn Sust.}$$

$$\text{Sust. mod} = \text{B Sgn Add} + \overline{\text{B Sgn}}\ \text{Sust.}$$

El resultado de la adición y de la sustracción de los módulos aparece en el acumulador. Si no hay arrastre más allá del peso de mayor orden del módulo, es decir en la posición A_0 del acumulador, el resultado aparece en valor real. Si no existe un arrastre en A_0 se necesita:

– cambiar el signo del acumulador,
– complementar a 2 el módulo del resultado,

"a condición de que el arrastre $A_0 = 1$ no señale un rebasamiento de capacidad del acumulador". Volveremos más adelante sobre este punto. Este complemento a 2 se obtiene por una transferencia del contenido del acumulador menos el signo al registro R, por una puesta a cero de las posiciones del módulo del acumulador, y luego por una sustracción del contenido del registro R del contenido del acumulador. La figura 3b representa el diagrama del desarrollo de una operación de adición de dos números. Obsérvese que habrá arrastres en A_0 en el caso de una adición de dos números de signos opuestos. Un rebasamiento de capacidad del acumulador (Overflow) se producirá cuando $A_0 = 1$ en el caso de la adición de dos números del mismo signo o si $A_0 = 1$ en el caso de la sustracción, de dos números de signos diferentes. Una báscula "Overflow" se pone entonces en 1, señalando un resultado inválido. Supongamos, para mayor comodidad de la exposición, que la operación se efectúa en tres etapas o tres ciclos: un ciclo de

instrucción que es común a todas las operaciones, un ciclo de preparación que prepara la ejecución de la operación y un ciclo de ejecución que ejecuta la operación. Esta clasificación es puramente arbitraria. Numerosas máquinas solamente utilizan dos ciclos: un ciclo de instrucción y un ciclo de ejecución. Se utilizará para cada ciclo un único distribuidor de tiempos. Este será generalmente un "anillo" que contendrá tantas básculas como tiempos elementales en el ciclo más largo. Así en el caso de la figura 3b, el ciclo más largo corresponde al ciclo de ejecución *E*. El distribuidor de tiempos dará cuatro tiempos elementales para cada uno de los ciclos. Algunos de ellos sólo serán utilizados para el ciclo de instrucción y de preparación. Los tres ciclos *I*, *P* y *E* se encadenan automáticamente. En el capítulo siguiente volveremos sobre este punto.

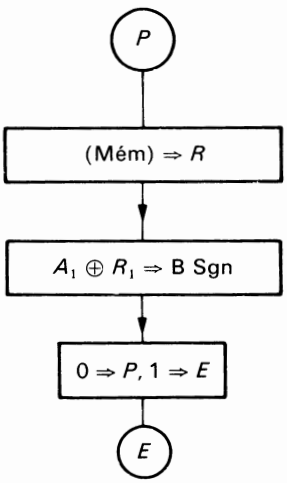


Fig. 3a. *Ciclo de preparación de la operación de adición y de sustracción*

Las figuras 3a, 3b, 3c representan el desarrollo de los ciclos de preparación y de ejecución de una adición y de una sustracción de dos números representados en valor absoluto y con signo.

No nos ocupamos aquí del ciclo de instrucción. Veremos en un capítulo posterior que la naturaleza del ciclo de instrucción puede influir sobre la realización de los ciclos de preparación y de ejecución, en lo que concierne al número de tiempos elementales dados por el distribuidor de tiempos. El paso del ciclo *P* al ciclo *E* se efectuará haciendo una puesta a cero de la báscula *P* y una puesta en 1 de la báscula *E*. Lo mismo ocurre al final del ciclo *E* para volver al ciclo *I* ($0 \Rightarrow E$, $1 \Rightarrow I$).

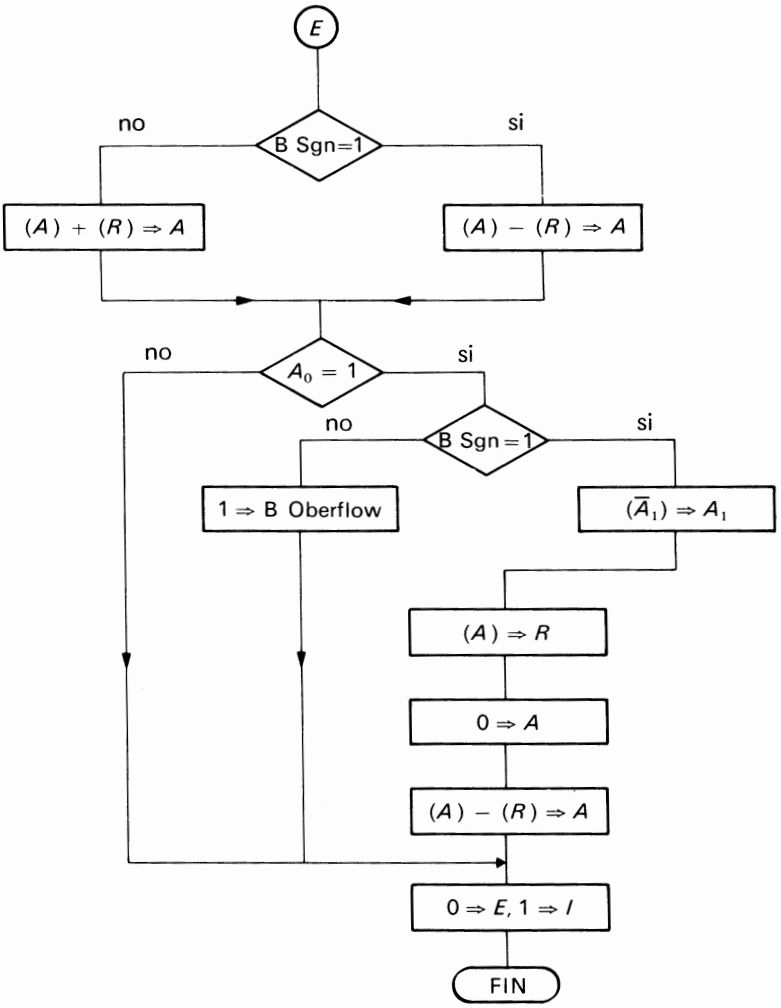


Fig. 3b *Ciclo de ejecución de la operación de adición. Obsérvese que la báscula B Overflow, puesta en 1 durante la operación señala un resultado invariable*

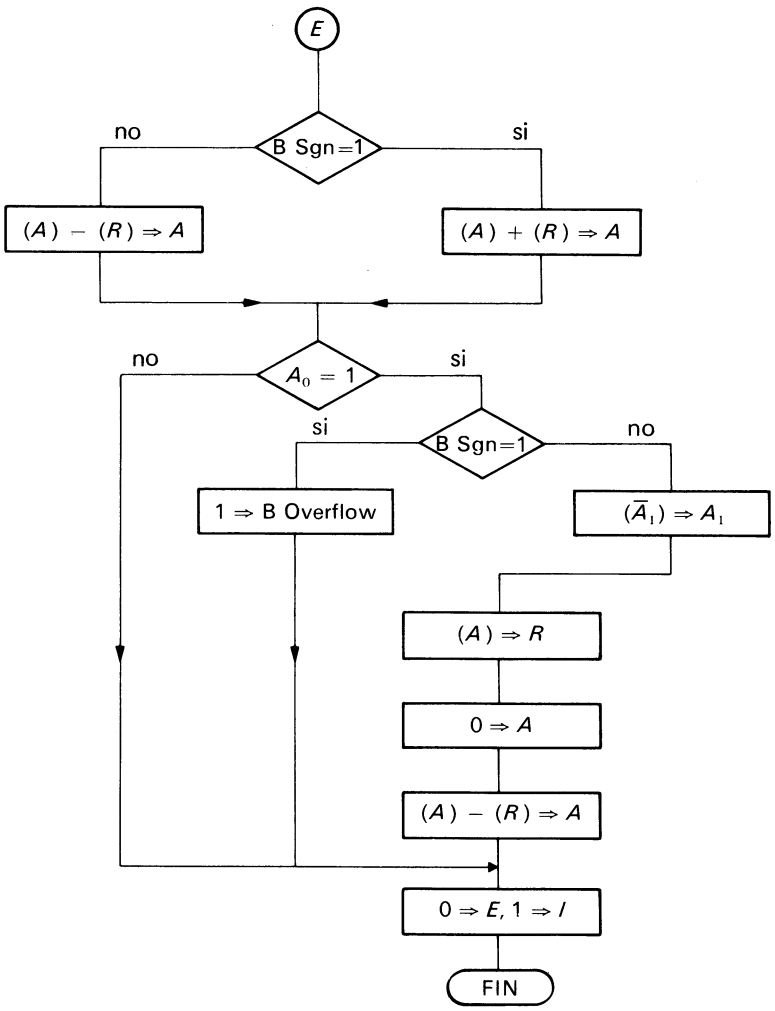


Fig. 3c. *Ciclo de ejecución de la sustracción. Obsérvese que la báscula B Overflow, puesta en 1 durante la operación, señala un resultado inválido*

El ciclo P es idéntico para las operaciones de adición y sustracción. Solamente el ciclo E es diferente según que se trate de una adición o de una sustracción. Los tres ciclos I, P y E son realizados por un distribuidor de tiempos controlado por el reloj del calculador (Fig. 4) asociado a tres básculas I, P y E. Cada secuencia de T_0 a T_4 efectuada por el distribuidor de tiempos es combinada con una de las básculas I, P y E para realizar los tres ciclos necesarios a la interpretación de la instrucción (I), la preparación (P) y la ejecución (E) de la operación contenida en la instrucción.

Fig. 4. *Distribuidor de tiempos para la realización de la adición y sustracción de dos números representado en valor absoluto y con signo*

El ciclo I es el mismo para todas las operaciones aritméticas, no lo representaremos. Por el momento nos limitaremos a estudiar los ciclos de preparación P y de ejecución E. Estos están representados en las figuras de 3a a 3c. La figura 5 representa la realización del ciclo P.

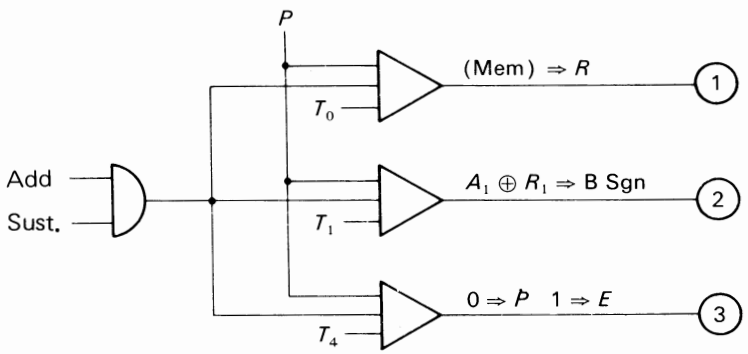


Fig. 5. *Realización de ciclo de preparación de la adición y de la sustracción de dos números representados en valor absoluto y con signo*

Obsérvese que los impulsos T_2 y T_3 del distribuidor de tiempos no se utilizan y que el paso del ciclo de preparación al ciclo de ejecución se efectúa en el tiempo T_4. La realización del ciclo de ejecución está representada en la figura 6. Utiliza todos los impulsos del distribuidor de tiempos.

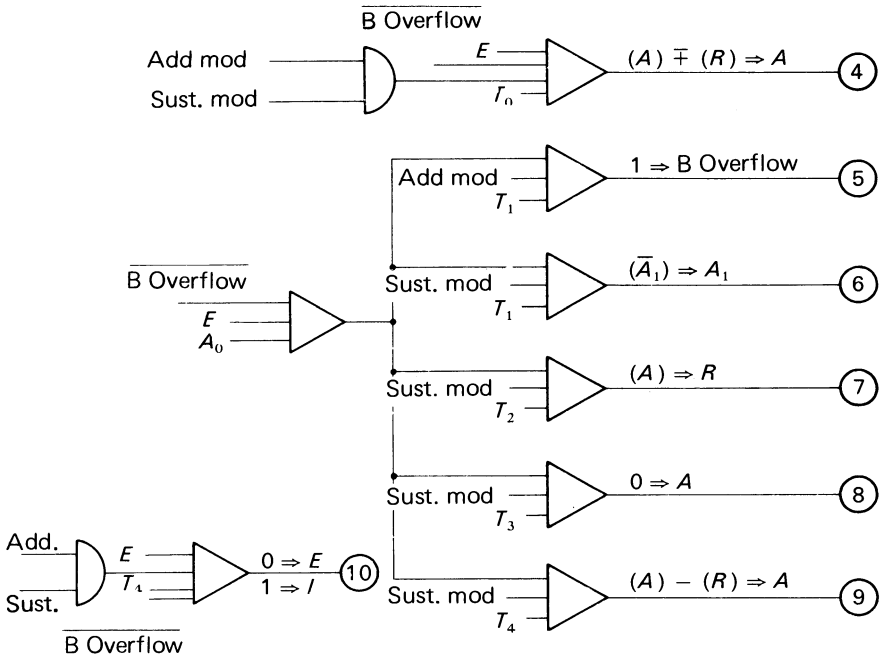


Fig. 6. *Realización del ciclo de ejecución de la adición y de la sustracción de dos números representados en valor absoluto y con signo*

Se termina en el tiempo T_4 que pone a cero la báscula E y pone en 1 la báscula I: volvemos así al ciclo de instrucción. El esquema general del adicionador-sustractor y de sus circuitos asociados está representado en la figura 7, en la que se han hecho aparecer todos los mandos del ciclo de preparación y de ejecución de la operación. Obsérvese en particular las básculas P y E de los ciclos de preparación y de ejecución, la báscula B Overflow de rebasamiento de capacidad del acumulador, la báscula B Sgn que memoriza la identidad o la desigualdad de los signos, el decodificador de operaciones que indica la naturaleza de la operación a ejecutar (adición o sustracción), el circuito de control de las operaciones que ordena la adición o la sustracción de los módulos según los signos de los operandos y la naturaleza de la operación a efectuar. Nótese que el resultado de la operación puede ser obtenido en T_0 cuando haya ausencia de arrastre en A_0. Por el contrario si $A_0 = 1$ y si estamos en presencia de una sustracción de módulos será necesario esperar a T_4 para obtener el resultado. El rebasamiento de capacidad es señalado por la báscula B Overflow cuando $A_0 = 1$ y que estemos en presencia de una adición de módulos.

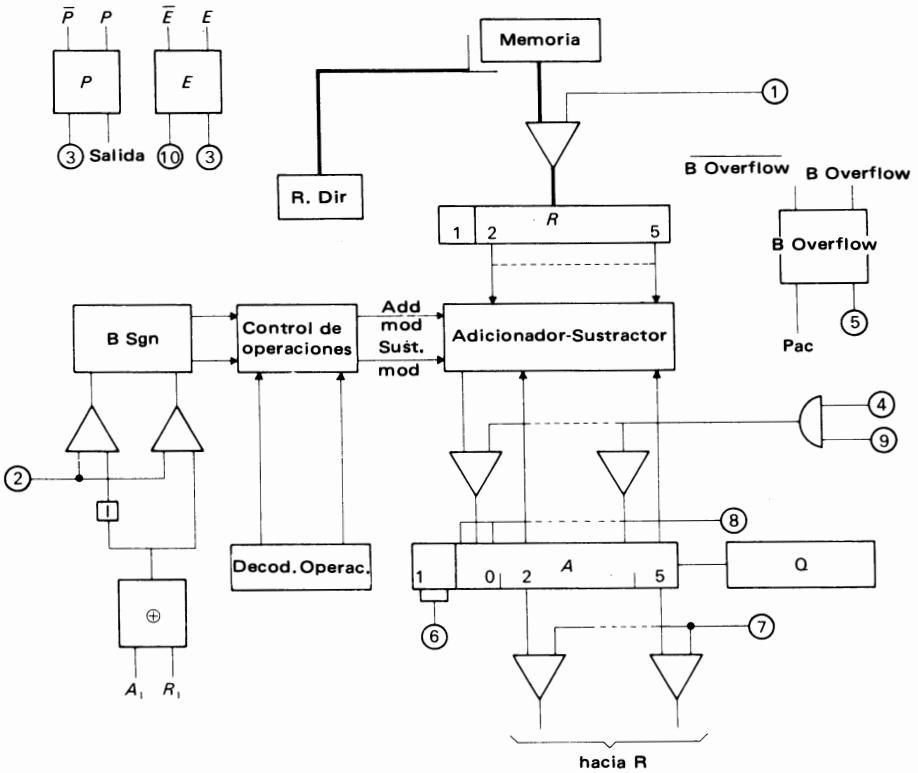


Fig. 7. *Adicionador-sustractor para números representados en valor absoluto y con signo*

4.1.2 *Números negativos representados por sus complementos a* 2

El tiempo de preparación y de ejecución es menor y solamente comprende tres tiempos elementales. El ciclo de preparación está representado en la figura 8.

La ejecución de la operación se efectúa en un solo tiempo, los otros dos tiempos son necesarios para la puesta en 1 eventual de una báscula Overflow y el retorno al ciclo *I* (fin Op.). El adicionador-sustractor efectúa, bajo una orden del decodificador de operación, la suma o la diferencia de los dos números peso por peso incluido el signo. El resultado aparece en valor real en el acumulador si éste es positivo, bajo forma de su complemento a 2 si éste es negativo. Estaremos en presencia de un rebasamiento de capacidad del acumulador, cuyo resultado es inválido, cada vez que se produzca un arrastre del módulo en el signo o un arrastre más allá del signo pero no los dos. Es decir:

$$\text{Overflow} = C_1 \oplus C_2 \,.$$

El lector encontrará en el anexo otro método muy utilizado.

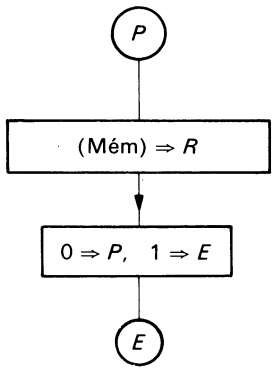


Fig. 8. *Ciclo de preparación de la adición y de la sustracción en el caso de números negativos representados por sus complementos a 2*

La figura 9 representa el desarrollo del ciclo de ejecución.

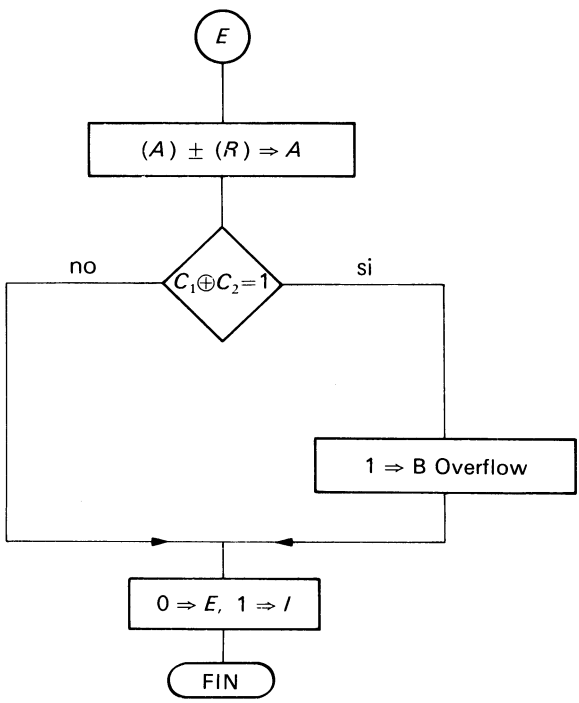


Fig. 9. *Ciclo de ejecución de la adición y de la sustracción de dos números representados en complemento a 2 cuando son negativos*

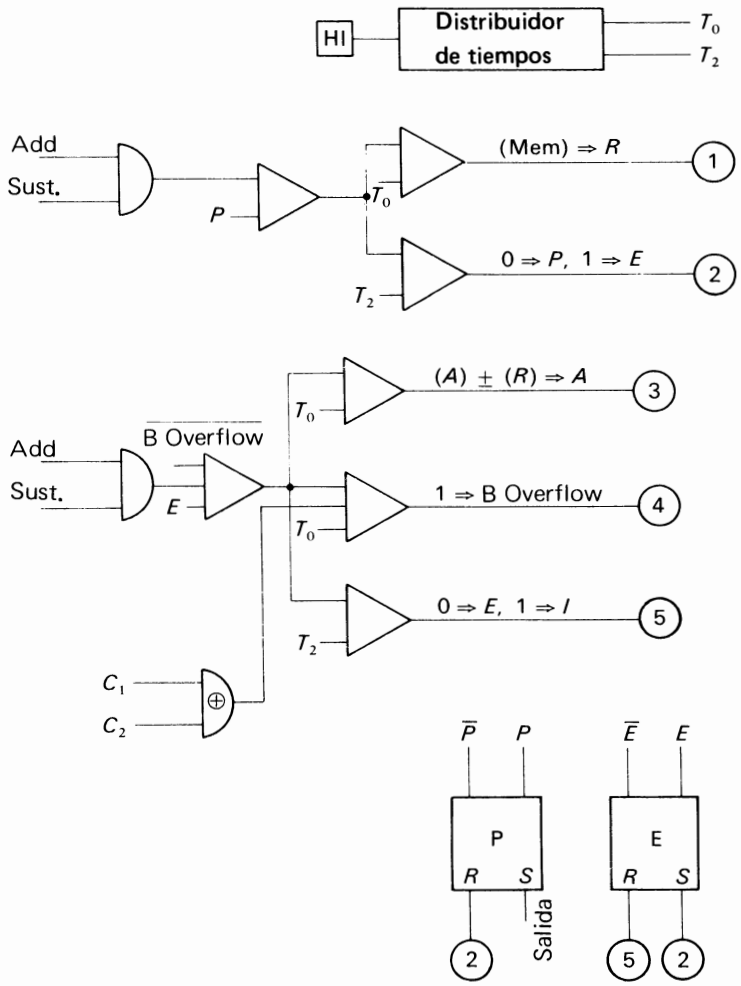


Fig. 10. *Realización de dos ciclos de preparación y de ejecución de una adición y una sustracción en las que los números están representados por sus complementos a 2*

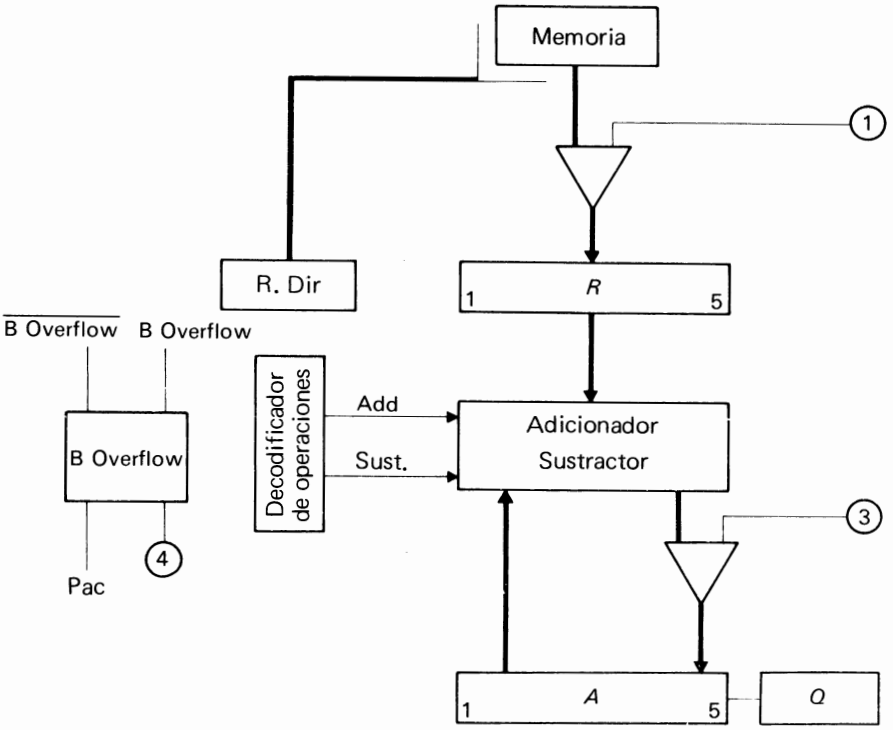


Fig. 11. *Adicionador-sustractor para números negativos representados por sus complementos a 2. El registro Q no se utiliza*

5. La multiplicación

La multiplicación de dos números de n dígitos da un resultado de $2n$ dígitos más el signo. Es pues necesario prever dos registros para contener el resultado de la multiplicación. Este será contenido en el acumulador y el registro Q. Luego, la unidad aritmética utiliza tres registros para la multiplicación:

– el registro R que contiene el multiplicando durante toda la multiplicación,
– el acumulador que efectúa cada suma parcial,
– el registro Q que contiene el multiplicador al principio de la operación.

El resultado final aparece en el conjunto de los dos registros acumulador y registro Q, este conjunto es designado por AQ. El multiplicador desaparece. Es necesario, además, prever un contador de desplazamientos C que cuenta el número de desplazamientos necesarios para la ejecución de la multiplicación y detiene el proceso al final de la operación.

5.1 Números representados en valor absoluto y con signo

La multiplicación de dos números representados en valor absoluto y con signo, se efectúa del mismo modo que a mano:

– multiplicar los valores absolutos de los dos números,

– tratar los signos separadamente.

Al principio de la operación el acumulador es puesto a cero, el multiplicando es introducido con su signo en el registro R, el número de desplazamiento en el contador C. El producto de los signos $Q_1 \oplus R_1$ se realiza antes de la ejecución de la operación y memorizado por la báscula B Sgn. El acumulador tiene una posición A_0 destinada a registrar un arrastre eventual en el curso de la realización de las sumas parciales. El conjunto acumulador-registro Q (salvo las posiciones A_1 y Q_1) es sometido a un desplazamiento de una posición a la derecha, a cada impulso de avance.

La multiplicación se efectúa del modo siguiente: según la naturaleza del peso de menor orden del registro Q se adiciona el contenido del registro R (multiplicando) al contenido del acumulador, o no se efectúa ninguna operación. El primer caso corresponde a $Q_5 = 1$, el segundo a $Q_5 = 0$. El contenido del acumulador y del registro Q (AQ) es seguidamente desplazado de una posición a la derecha, y el contenido del contador de desplazamientos disminuye en una unidad. El peso de menor orden del multiplicador se pierde. Q_5 contiene ahora el siguiente peso de menor orden del multiplicador. El proceso continua del mismo modo hasta que el contenido del contador de desplazamientos sea nulo. Entonces se transfiere el signo del resultado a A_1. El resultado de la multiplicación aparece en el conjunto acumulador-registro Q, el peso de mayor orden en el acumulador, el peso de menor orden en el registro Q, el signo en A_1. *No existe rebasamiento de capacidad*

Micro - operaciones	A_0	A_2	A_3	A_4	A_5	Q_2	Q_3	Q_4	Q_5	(C)	A_1	Q_1
Principio de operación	0	0	0	0	0	1	1	0	1	4	0	1
$(A) + (R) \Rightarrow A$	0	1	0	1	1	1	1	0	1	4	0	1
$(AQ_i) \Rightarrow AQ_{i+1}$	0	0	1	0	1	1	1	1	0	3	0	1
$(AQ_i) \Rightarrow AQ_{i+1}$	0	0	0	1	0	1	1	1	1	2	0	1
$(A) + (R) \Rightarrow A$	0	1	1	0	1	1	1	1	1	2	0	1
$(AQ_i) \Rightarrow AQ_{i+1}$	0	0	1	1	0	1	1	1	1	1	0	1
$(A) + (R) \Rightarrow A$	1	0	0	0	1	1	1	1	1	1	0	1
$(AQ_i) \Rightarrow AQ_{i+1}$	0	1	0	0	0	1	1	1	1	0	0	1
$R_1 \oplus Q_1 \Rightarrow A_1$	0	1	0	0	0	1	1	1	1	0	1	0

Fig. 12. *Multiplicación en valor absoluto y con signo de X = 01011 e Y = 11101*

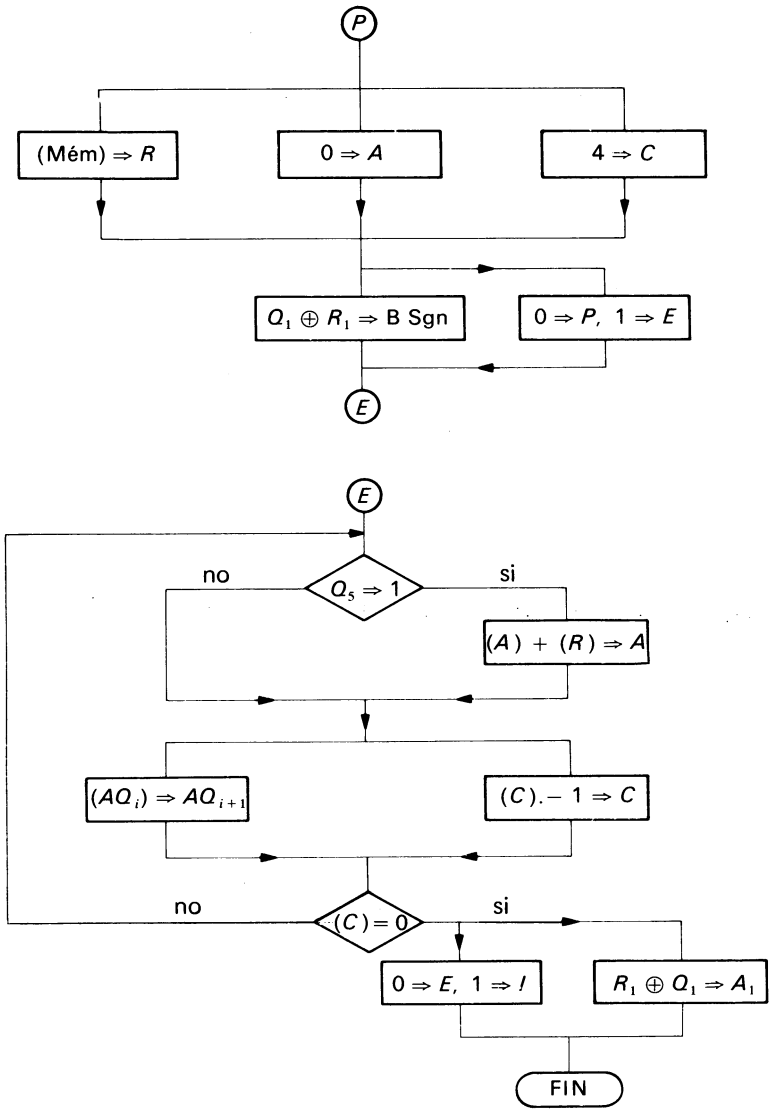


Fig. 13. *Ciclos de preparación y de ejecución de la multiplicación de dos números representados en valor absoluto y con signo*

en la multiplicación. Al final de la operación el multiplicador se ha perdido. La figura 12 da un ejemplo de la multiplicación (multiplicando = X, multiplicador = Y).

La figura 13 representa el diagrama de los ciclos de preparación y de ejecución de la multiplicación. Obsérvese que la ejecución de la multiplicación puede contener varios ciclos E. El último ciclo tendrá lugar cuando $(C) = 0$. El distribuidor de tiempos emite, como anteriormente, tres tiempos elementales de T_0 a T_2.

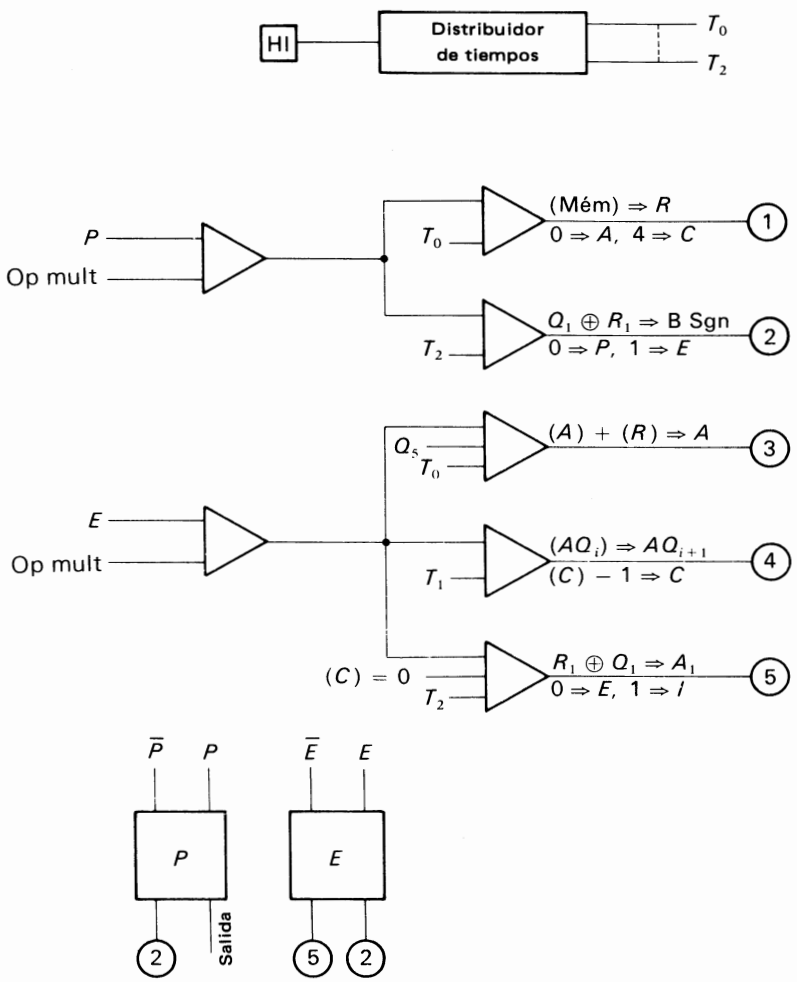


Fig. 14. *Realización de dos ciclos de preparación y de ejecución de la multiplicación de dos números expresados en valor absoluto y con signo*

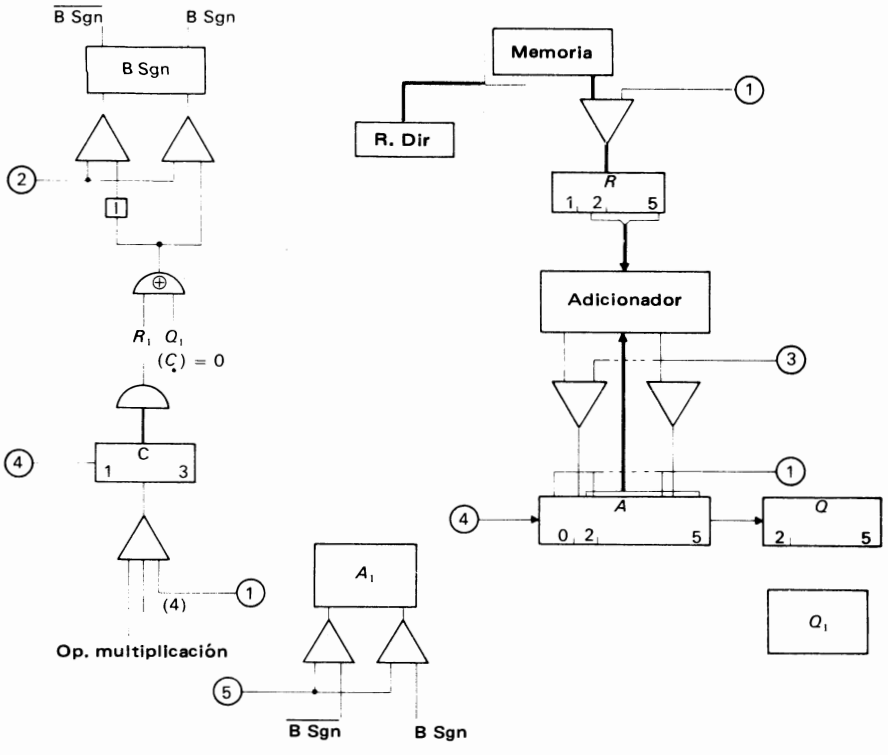


Fig. 15. *Multiplicador para números representados en valor absoluto y con signo*

5.2 El multiplicador de Booth

Si los números a multiplicar están representados por sus complementos a 2 cuando son negativos, se utilizará el algoritmo de Booth para realizar la multiplicación. Este ha sido detallado en el capítulo 3, del tomo I. Como el anterior, comprende:

- un acumulador A asociado con un registro Q y una báscula suplementaria Q_s,
- un registro R que contiene el multiplicando,
- un adicionador-sustractor que efectúa la suma o la diferencia del multiplicando y del contenido del acumulador,
- un contador de desplazamientos,
- un circuito de control de cada operación elemental que suministra un mando de adición o de sustracción según la naturaleza de los dígitos contenidos en Q_5 y Q_s.

El conjunto acumulador-registro Q-báscula Q_s es sometido a un desplazamiento

a la derecha con mantenimiento del signo después de cada operación elemental. Cuando el contenido del contador de desplazamientos es nulo, la multiplicación ha terminado. El resultado aparece en el acumulador y el registro Q. Si hay rebasamiento de capacidad durante la multiplicación (multiplicando =10000) el signo del acumulador es invertido en el momento del desplazamiento. A_1 contiene el signo del resultado. La última operación consiste en poner a cero Q_s. El ciclo de preparación consiste en la introducción del multiplicando con su signo en el registro R, en poner a cero el acumulador y la báscula Q_s y en introducir en C el número de desplazamientos a realizar. Este es igual a n, si n es el número de dígitos del multiplicando y del multiplicador.

Al principio de la operación el multiplicador está en Q y $Q_s = 0$, de tal forma que la primera operación no puede ser otra cosa que una sustración si $Q_5 = 1$ o un simple desplazamiento de una posición a la derecha si $Q_5 = 0$. Las operaciones siguientes son impuestas por el contenido de Q_5 y Q_s según el cuadro de la figura 16. Estaremos en presencia de una adición cuando:

$$\text{Adición} = \overline{Q}_5 \, Q_s$$

y de una sustracción:

$$\text{Sustracción} = Q_5 \, \overline{Q}_s \, ,$$

de donde se obtiene la condición de adición o sustracción:

$$\text{Adición o sustracción} = Q_5 \oplus Q_s \, .$$

Inversamente no tendremos ninguna operación cuando:

$$\text{Sin operación} = \overline{Q_5 \oplus Q_s} = Q_5 \odot Q_s \, .$$

Q_5	Q_S	Operaciones
0	0	Dpz dr
0	1	Add Dpz dr
1	0	Sust Dpz dr
1	1	Dpz dr

Fig. 16. *Algoritmo de Booth*

Los ciclos de preparación y de ejecución de la multiplicación están representados en la figura 17. Obsérvese la función preponderante del contador de desplazamientos C, que impone la continuación del ciclo E hasta que su contenido es nulo. La realización de estos ciclos está representada en la figura 18. La figura 19 representa la realización del multiplicador y la figura 20 el detalle de las operaciones. La operación de desplazamiento es descrita en el anexo.

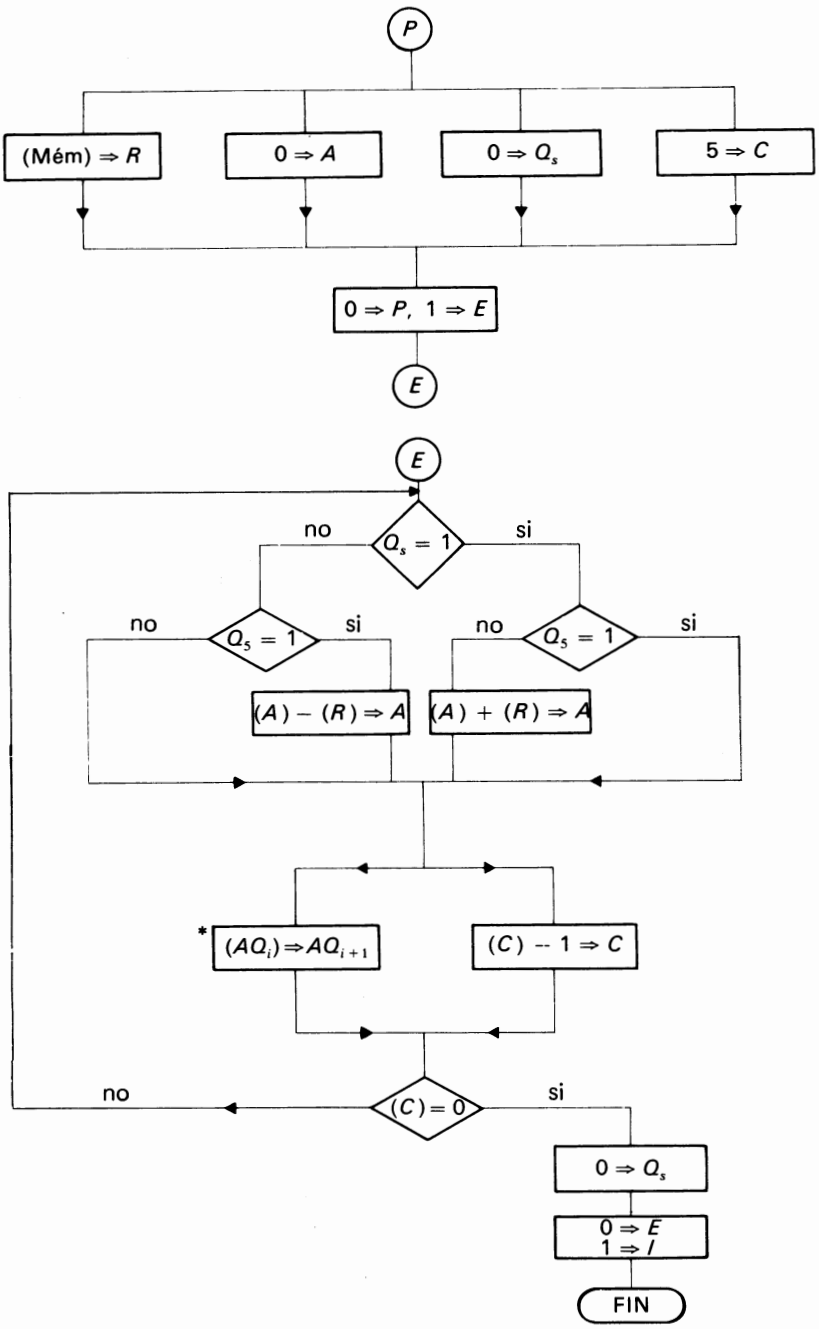


Fig. 17. *Ciclos de preparación y de ejecución de la multiplicación según el algoritmo de Booth (*) Ver anexo*

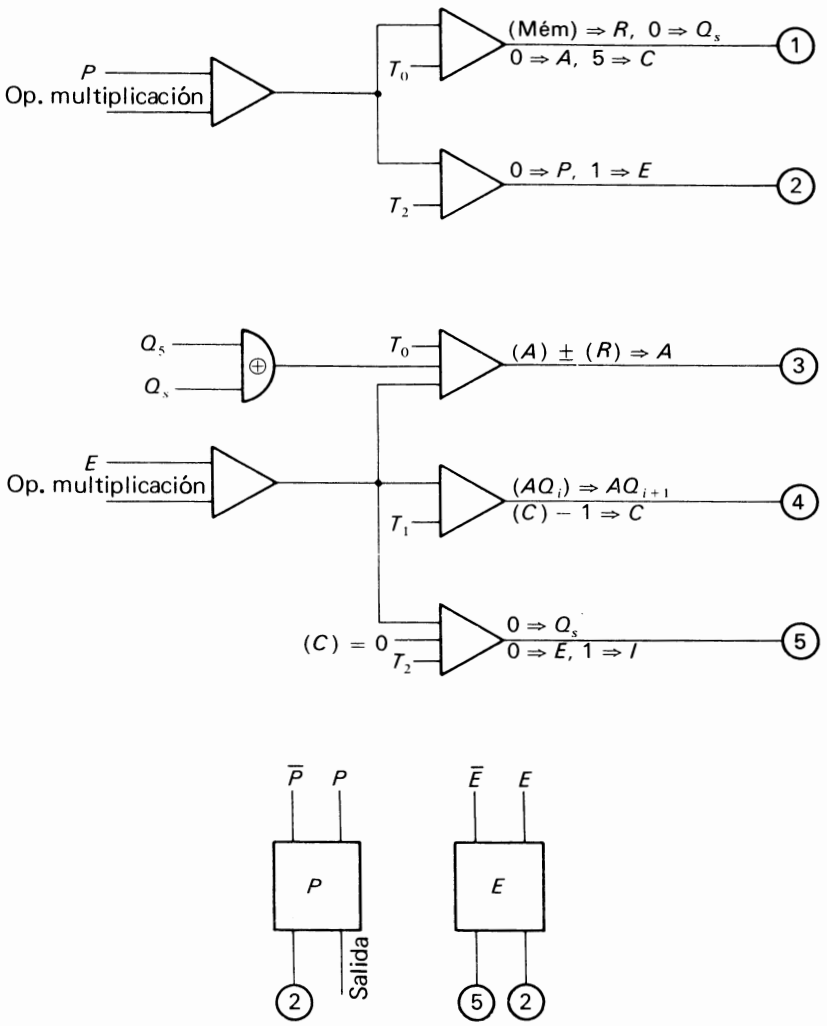


Fig. 18. *Realización de dos ciclos de preparación y de ejecución de la multiplicación según el algoritmo de Booth*

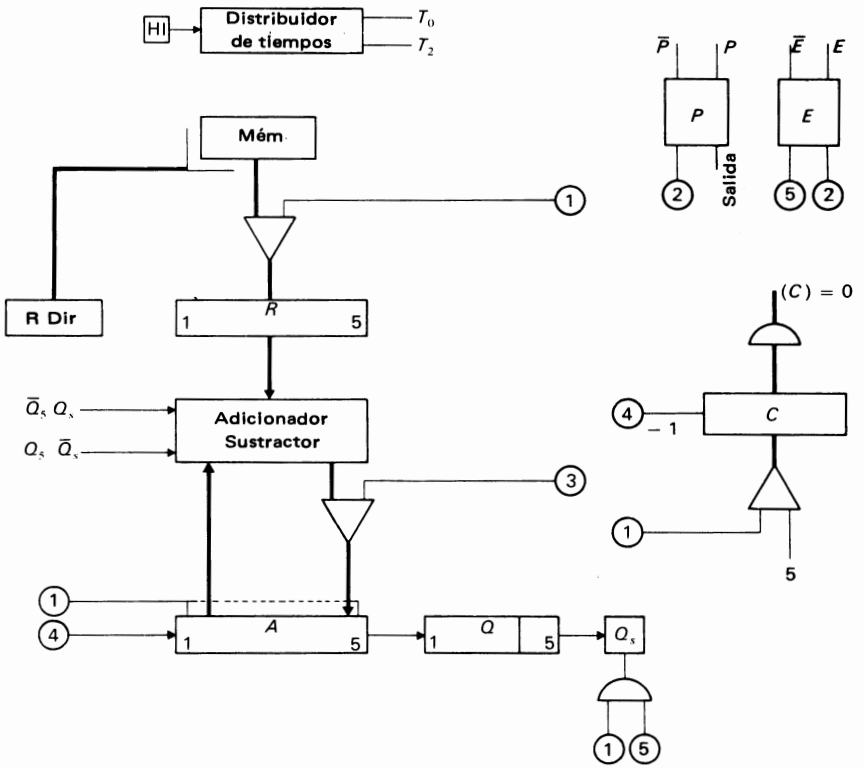


Fig. 19. *Multiplicador para la utilización del algoritmo de Booth*

5.3 Multiplicador de Booth, segunda versión

La báscula adicional Q_s puede suprimirse utilizando la variante del algoritmo de Booth estudiado en el capítulo 3 del tomo I. En este caso hay que añadir al multiplicador de la figura 19 una báscula de operación B Op que memoriza la operación anterior a la que se está realizando.

El número de desplazamientos es igual al número de bits de cada uno de los dos números y el resultado final aparece entre A_1 y Q_5. Los desplazamientos se efectúan conservando el signo del acumulador si no hay rebasamiento de capacidad. En caso contrario, el signo del acumulador es invertido en el momento del desplazamiento. Esta operación es descrita en el anexo.

La ejecución de la operación varía según el contenido de la báscula B Op y del último peso Q_5 del registro Q. Si $Q_5 \oplus$ B Op $= 1$, se desplaza el contenido del acumulador y del registro Q de una posición a la derecha disminuyendo en una unidad el contenido del contador de desplazamientos C. Si $Q_5 \odot$ B Op $= 1$, se

efectuará una adición o una sustracción del multiplicando según que $Q_4 = 0$ o que $Q_4 = 1$, a condición de que el contenido de C sea diferente de 1. Las operaciones sucesivas están definidas en el cuadro de la figura 21 que volvemos a reproducir para mayor facilidad. Al principio de la multiplicación se supone que la operación anterior es una adición (1 ⇒ B Op).

Micro-operaciones	Acumulador	Registro Q				Q_S	Micro-operaciones siguientes	0
		Q_1	Q_2	Q_3	Q_4	Q_5		
	0 | 0 0 0 0	1	1	0	1	1	0	Sust X, Dpz dr	5
	1 | 0 1 0 1							
$(A) - (R) \Rightarrow A$	0 | 1 0 1 1	1		0	1	1	0		
$(AQ_i) \Rightarrow AQ_{i+1}$	0 | 0 1 0 1	1		1	0	1	1	Dpz dr	4
$(AQ_i) \Rightarrow AQ_{i+1}$	0 | 0 0 1 0	1		1	1	0	1	Add X, Dpz dr	3
	1 | 0 1 0 1							
$(A) + (R) \Rightarrow A$	1 | 0 1 1 1	1		1	1	1	1		
$(AQ_i) > AQ_{i+1}$	1 | 1 0 1 1	1		1	1	1	0	Sust X, Dpz dr	2
	1 | 0 1 0 1							
$(A) - (R) \Rightarrow A$	0 | 0 1 1 0	1		1	1	1	0		
$(AQ_i) \Rightarrow AQ_{i+1}$	0 | 0 0 1 1	0		1	1	1	1		1
$(AQ_i) \Rightarrow AQ_{i+1}$	0 | 0 0 0 1	1		0	1	1	1	Resultado	0

Fig. 20. *Diferentes fases de la multiplicación de X = 1/0101 por Y = 1/1011 X es el multiplicando, Y el multiplicador. Al principio de la operación X está en R. Obsérvese la función de Q_5 Q_s en la definición de las operaciones sucesivas. La multiplicación termina cuando el contenido del contador de desplazamientos es nulo (C) = 0. Esta figura representa un ejemplo de multiplicación de los números X = 10101 e Y = 11011*

Bits del multiplicador	Operación anterior	Nueva operación
0	Adición	Desplazamiento a la derecha
0 1	"	Adición, desplazamiento derecha
1 1	"	Sustracción, desplaz. derecha
0 0	Sustracción	Adición, desplazamiento derecha
1 0	"	Sustracción, despl. derecha
1	"	Desplazamiento derecha

Fig. 21. *Algoritmo de Booth, segunda versión*

Si $C = 1$, Q_4 contiene el peso de menor orden de la multiplicación, por lo tanto no puede ser tomado en consideración. En este momento si $Q_5 = 1$ y si la operación anterior es una adición se sustrae el multiplicando del contenido del acumulador. Si $Q_5 = 0$ y si la operación anterior es una sustracción se adiciona el multiplicando al contenido del acumulador. Esta última operación es seguida de un desplazamiento. En todos los casos cuando $(C) = 0$ la multiplicación ha terminado. La figura 22 representa el distribuidor de tiempos, las figuras 23 y 24 el ciclo de preparación y de ejecución de la multiplicación, las figuras 25, 26 y 27, la realización de estos ciclos y el esquema general del multiplicador.

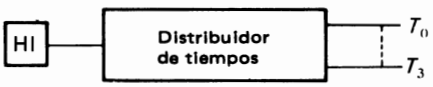


Fig. 22. *Distribuidor de tiempos del multiplicador de Booth (segunda versión)*

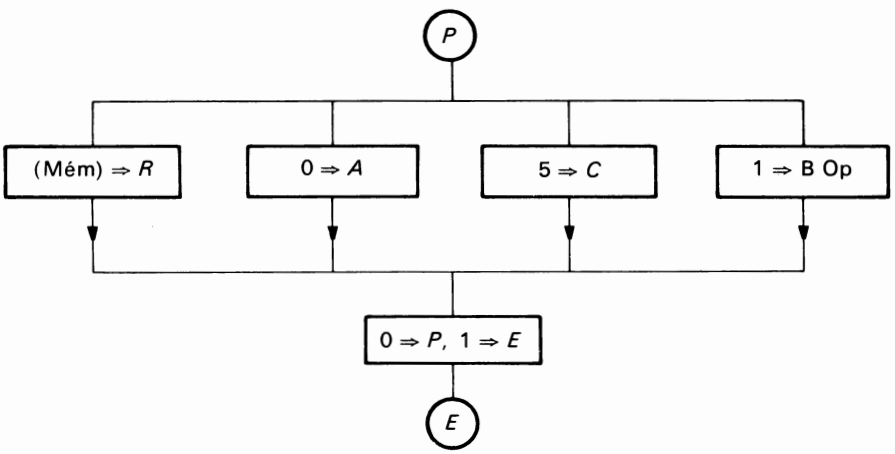


Fig. 23. *Ciclo de preparación de la multiplicación por el algoritmo de Booth (segunda versión)*

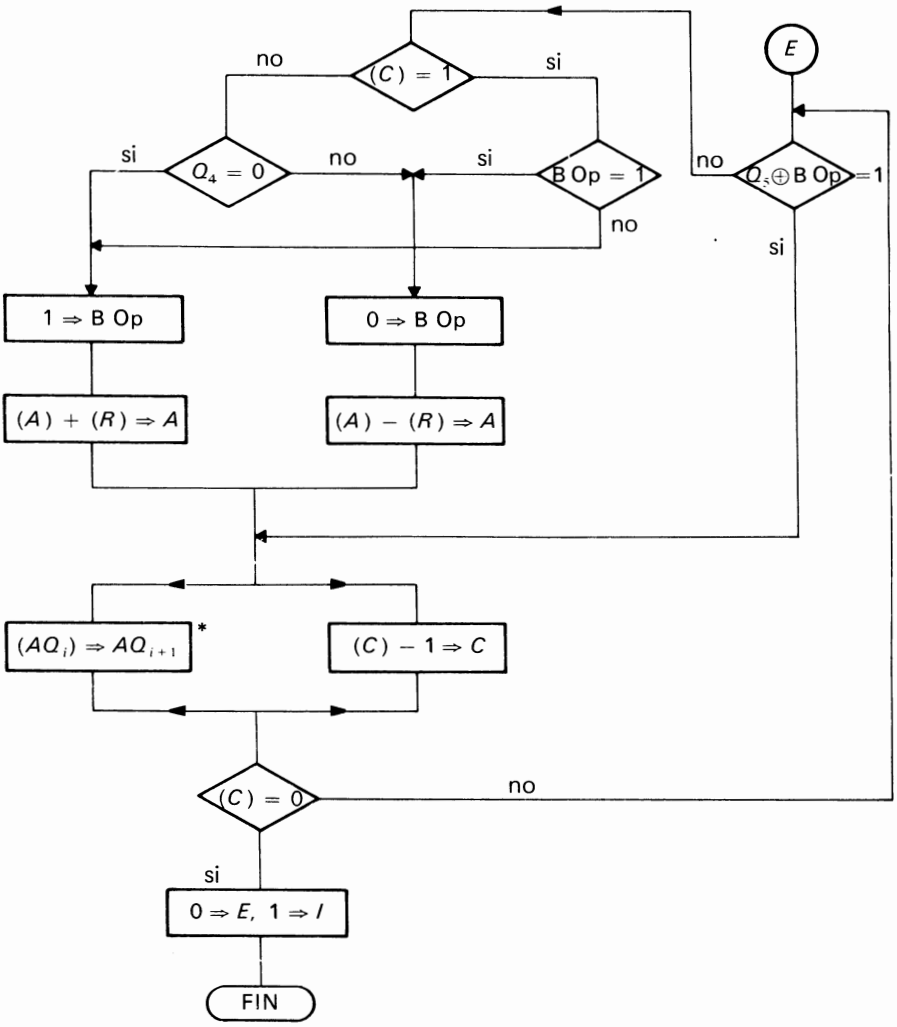


Fig. 24 *Ciclo de ejecución E de la multiplicación, por el algoritmo de Booth (segunda versión). (*) Ver anexo*

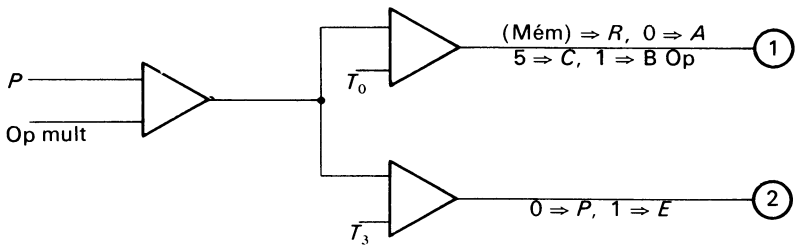


Fig. 25. *Realización del ciclo de operación de la multiplicación por el algoritmo de Booth (segunda versión).*

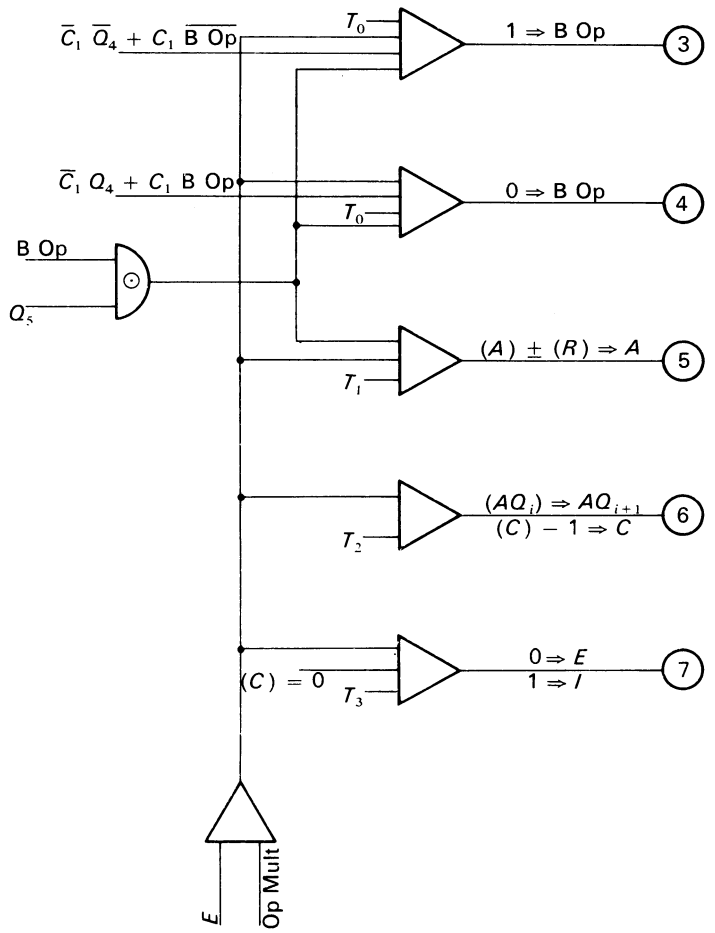


Fig. 26. *Realización del ciclo de ejecución E de la multiplicación (algoritmo de Booth, segunda versión)*

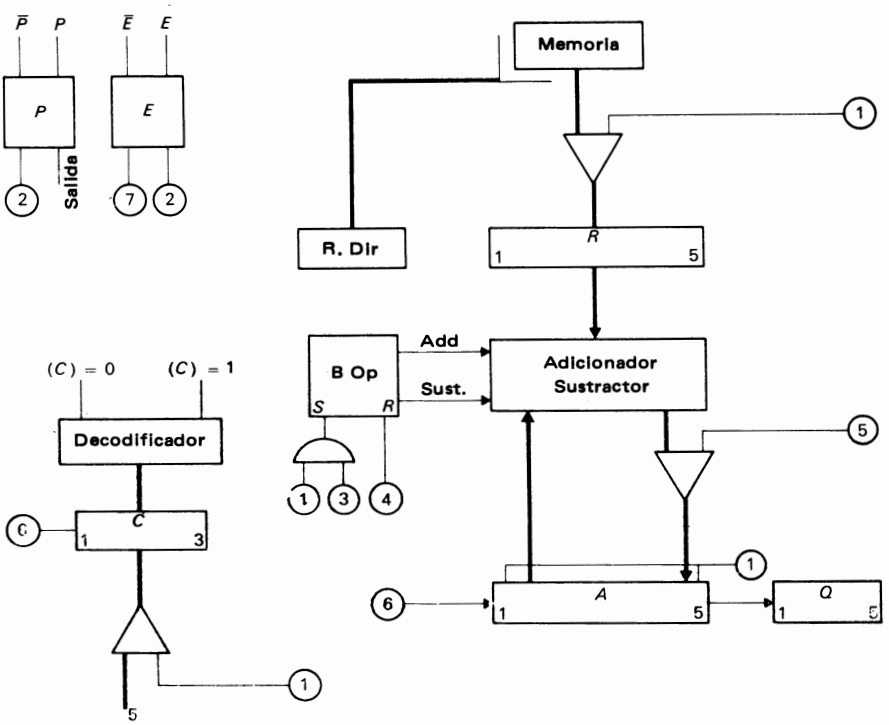


Fig. 27. *Multiplicador de Booth (segunda versión)*

5.4 Multiplicador rápido de productos simultáneos

Es posible realizar un multiplicador que elabora simultáneamente todos los productos parciales y los desplazamientos necesarios. El resultado de la multiplicación es obtenido inmediatamente en un tiempo igual al de la adición: se trata pues de un procedimiento muy rápido. El esquema de este multiplicador está representado en la figura 28. Obsérvese el gran número de adicionadores que es necesario utilizar en el caso de una multiplicación de números muy grandes. Por ello este procedimiento es poco empleado. Este método solamente se puede utilizar en el caso en que los números estén expresados en valor absoluto y con signo. Este es tratado separadamente, y el signo del resultado es transferido al final de la multiplicación a la posición signo del acumulador. La sucesión de las operaciones y la serie de adiciones con arrastre está representada a continuación (Fig. 29).

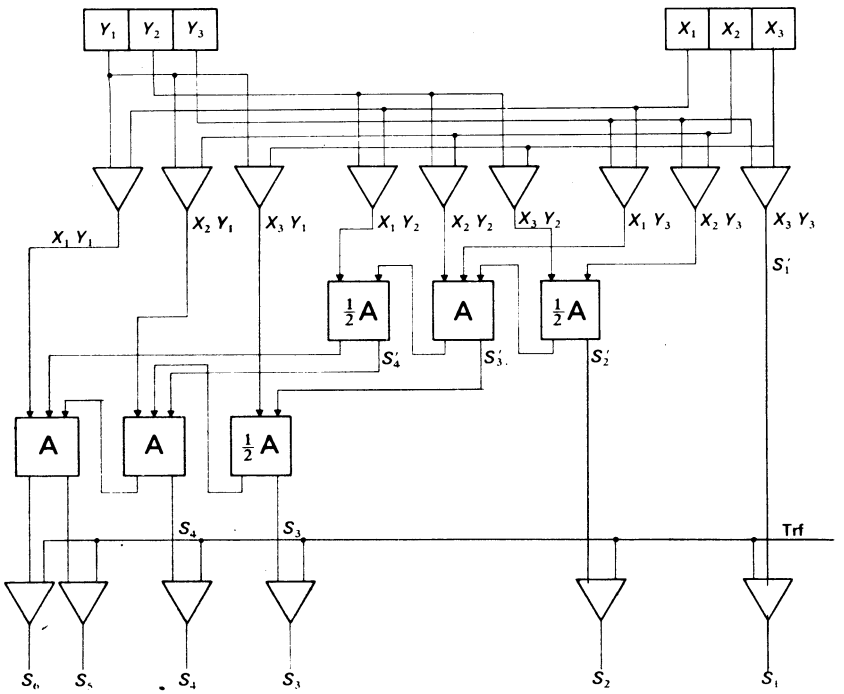


Fig. 28. *Esquema general del multiplicador rápido de productos simultáneos*

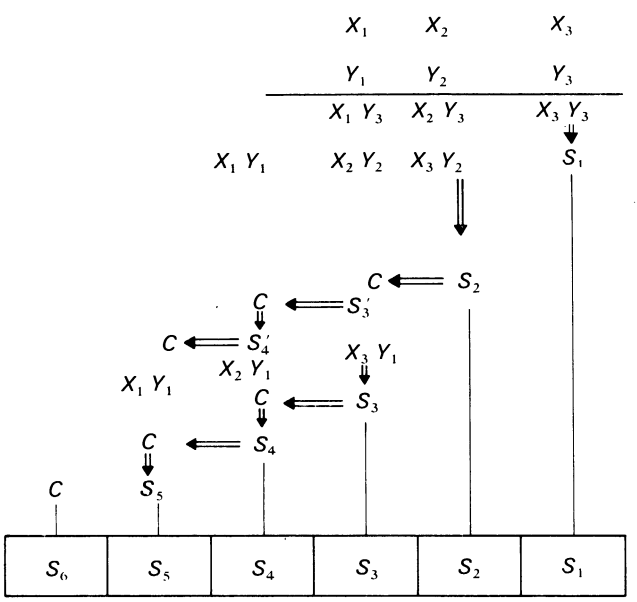


Fig. 29. *Detalle del funcionamiento del multiplicador de productos simultáneos*

6. La división

Vimos en el tomo I que existen dos métodos de división frecuentemente utilizados: el método que utiliza la restauración de los restos parciales y el método de comparación de los signos de los restos parciales y del signo del divisor. En el primero, los números se expresan en valor absoluto y con signo, en el segundo los números negativos son representados por sus complementos a dos.

6.1 División por el método de restauración de los restos parciales

Como anteriormente, la unidad aritmética comprende un acumulador A asociado a un registro Q. El acumulador tiene, como en el caso de la adición y de la sustracción de dos números expresados en valor absoluto y con signo, una posición suplementaria A_0 destinada a memorizar un arrastre eventual que rebase el módulo del resultado. Los registros A y Q (menos el signo) pueden desplazar simultáneamente sus contenidos de una posición a la izquierda. El registro R contiene el divisor durante toda la división. Una báscula operación "B Op" memoriza la operación a efectuar después de examinar el signo de los restos parciales. El dividendo es colocado en el acumulador y el registro Q antes de la operación. El número de desplazamientos necesarios para efectuar la división, es introducido en el contador de desplazamientos C al principio de la operación. Si el divisor es nulo, la división no es efectuada. El signo del resultado $A_1 \oplus R_1$ es elaborado antes de comenzar la división en un circuito "O exclusivo" y memorizado en una báscula B Sgn. La báscula B Op es posicionada sobre la sustracción ($0 \Rightarrow$ B Op) y se efectúa la sustracción del módulo del divisor del peso de mayor orden del módulo del dividendo $(A) - (R) \Rightarrow A$. Si el signo del resultado de la sustracción es positivo ($A_0 = 0$), una báscula B Op inv es puesta a 1: efectivamente el resultado es inválido (rebasamiento de capacidad del registro Q). Para que la división pueda efectuarse, el primer resto (resultado de la primera operación) debe obligatoriamente ser negativo ($A_0 = 1$). En este caso la báscula de operación B Op es posicionada sobre adición ($1 \Rightarrow$ B Op) y se restaura el dividendo por adición del divisor. El signo de cada resto parcial impone la naturaleza de la operación siguiente así como el dígito correspondiente del cociente.

Si $A_0 = 0$ el dígito del cociente es 1 y la operación siguiente a ejecutar, *después de un desplazamiento a la izquierda de una posición del conjunto acumulador-registro* Q es una sustracción. Si $A_0 = 1$, el dígito del cociente es 0, *se detiene el desplazamiento* y la operación siguiente es una adición (restauración del resto precedente). Después de la restauración del resto, no hay registro de un dígito del cociente, y el conjunto acumulador-registro Q es desplazado de una posición a la izquierda. Se sigue de este modo hasta que el contenido del contador de

desplazamientos sea nulo. El acumulador contiene entonces el último resto y el registro Q el cociente. Si el último resto es negativo se restaura el contenido del acumulador. Este contendrá entonces el resto exacto de la división. El signo del resto es el mismo que el del dividendo. La figura 30 representa el ciclo de preparación, la figura 31 el ciclo de ejecución de la división. Obsérvese la misión de la báscula operación inválida "B Op inv".

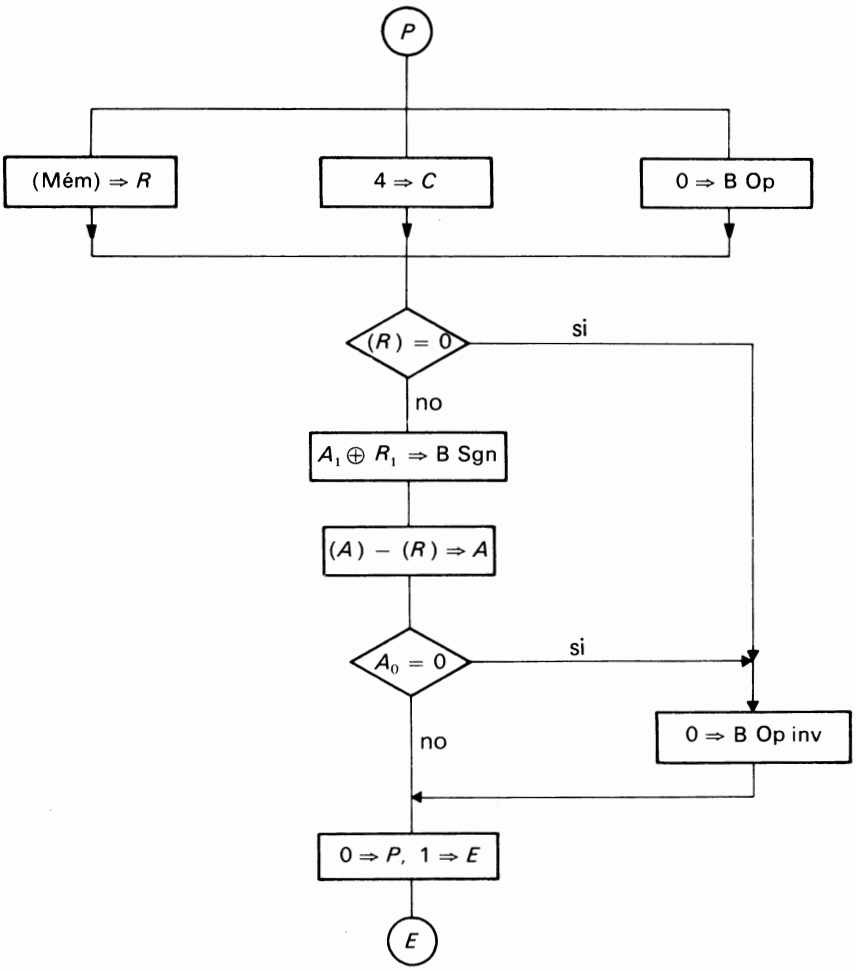


Fig. 30. *Ciclo de preparación de la división por el método de restablecimiento de los restos*

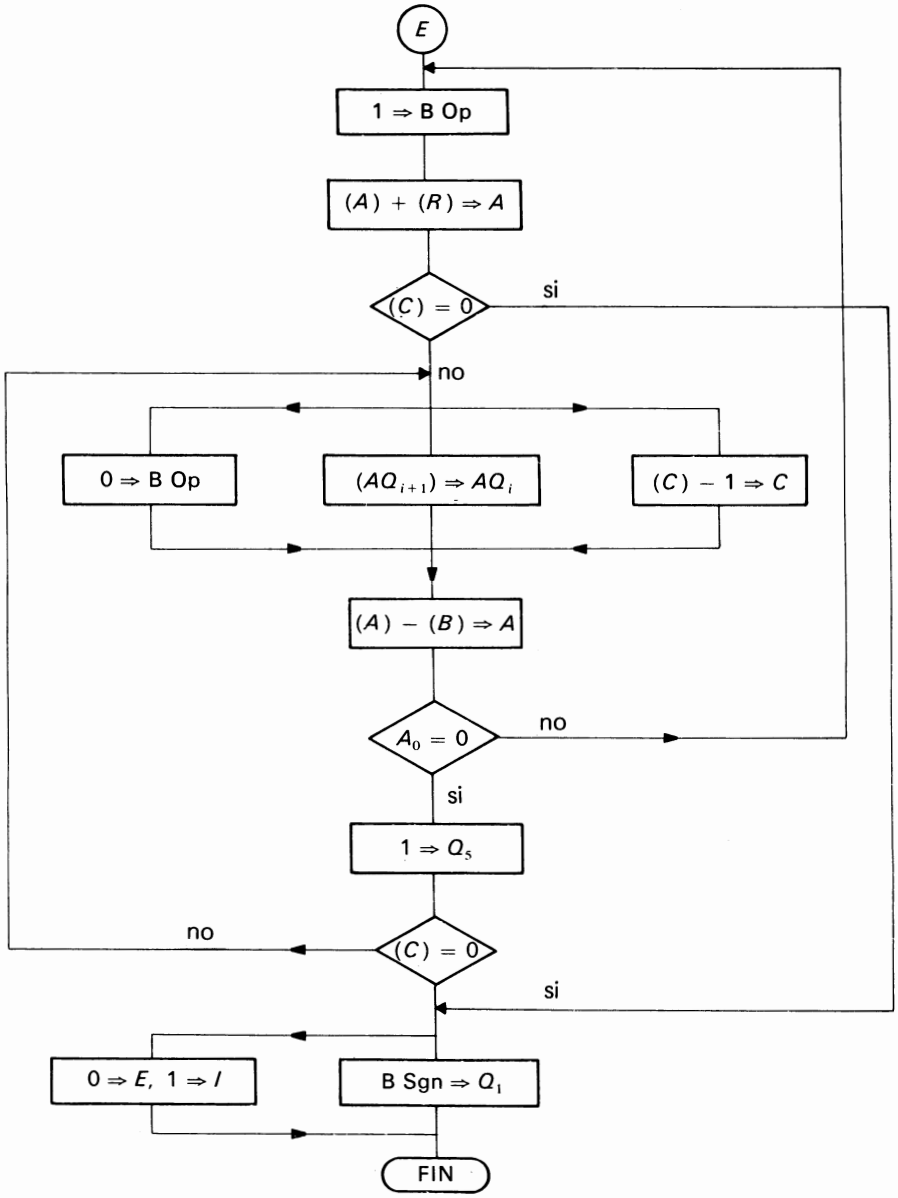


Fig. 31. *Ciclo de ejecución de la división por el método de restablecimiento de restos. Este ciclo se divide en dos ciclos, E_1 y E_2 para obtener la ejecución de la división*

Supongamos ahora, que la detección de una operación inválida en el ciclo de preparación no detiene el paso al ciclo E. Este será ejecutado, pero las micro-operaciones que se realizan en el caso de una operación válida, no se realizan en el caso de una operación inválida. Se observará que esto representa una pérdida de tiempo. En ciertas máquinas, la detección de una operación inválida en el curso del ciclo envía inmediatamente al ciclo de instrucción siguiente sin terminar el que está en curso de ejecución. No es éste el caso que nos ocupa. Si una división inválida es detectada durante el ciclo de preparación, este ciclo será ejecutado enteramente así como el ciclo de ejecución, pero las micro-operaciones del ciclo de ejecución serán eliminadas por la báscula "B Op inv". La figura 32 representa el esquema de realización del ciclo de preparación de la división. La puesta a cero de la báscula B Op inv se efectuará después de un test programado sobre su contenido.

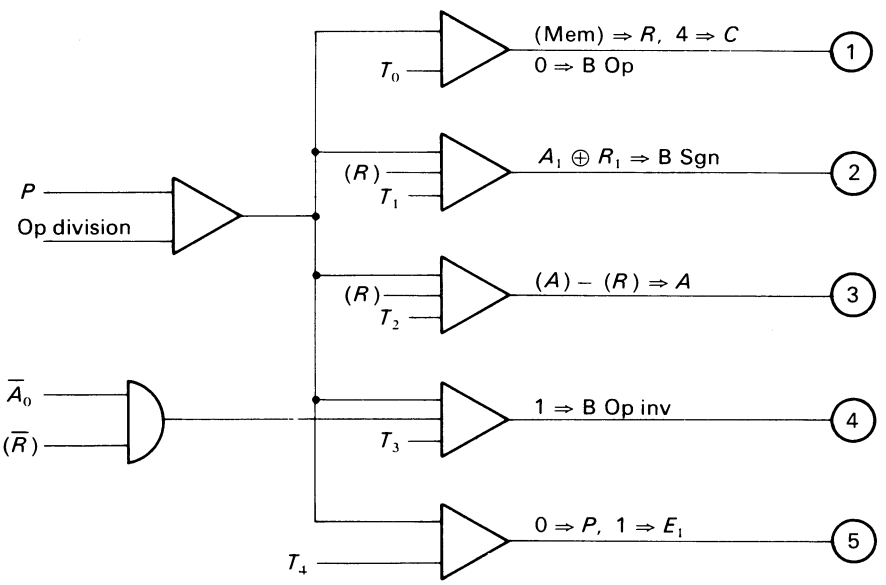


Fig. 32. *Realización del ciclo de preparación de la división por el método del restablecimiento de los restos parciales*

La figura 33 representa el ciclo de ejecución de la división por el método de restauración de los restos parciales. Obsérvese la distribución en dos ciclos, E_1 y E_2, del ciclo de ejecución E.

En estas figuras los tiempos T_0 a T_4 tienen lugar a condición que la operación sea válida. Puede observarse que estos impulsos son eliminados si B Op inv = 1 (Fig. 34). El esquema general de la división está representado en la figura 35. Se obtiene

una solución más simple, no transmitiendo el resultado de la sustracción al acumulador cuando éste es negativo. De este modo se evita la restauración de los restos parciales por adición del divisor. La conexión adicionador-sustractor acumulador será eliminada cuando el arrastre que rebase el módulo del resultado será igual a 1.

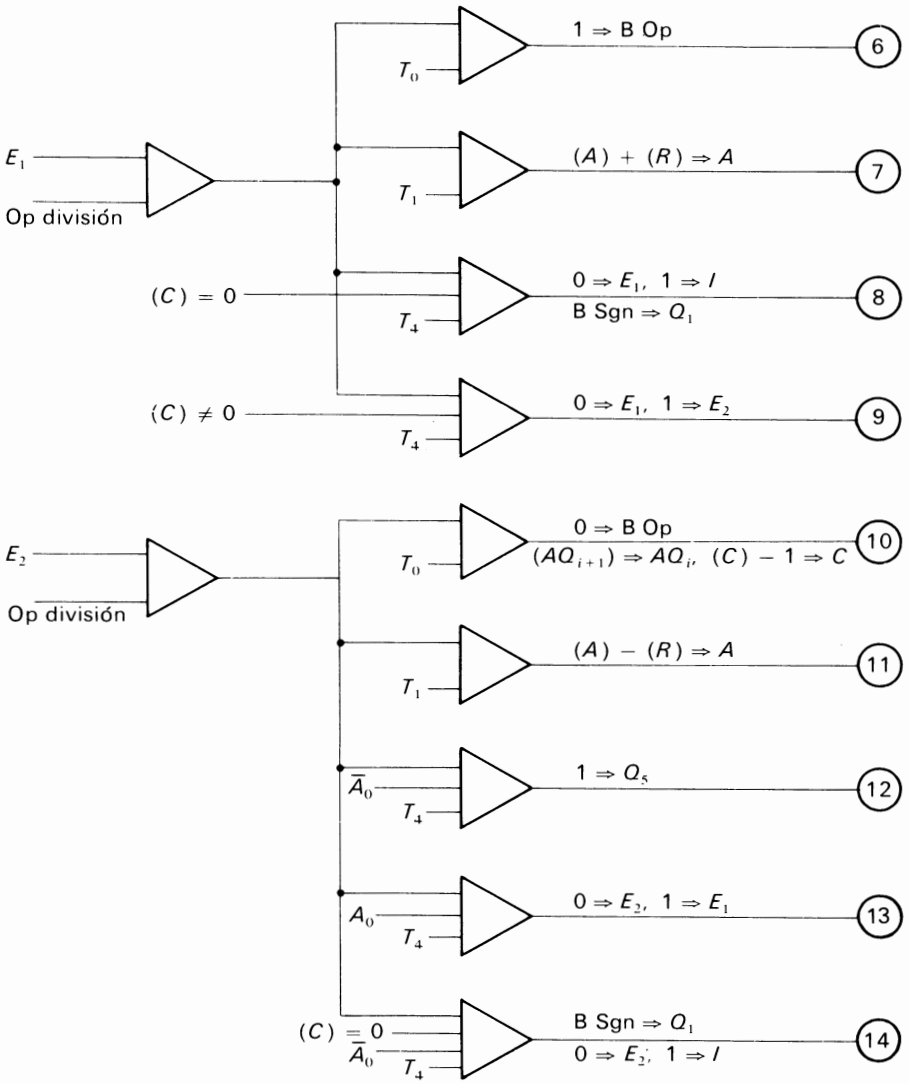


Fig. 33. *Realización del ciclo de ejecución de la división por el método del restablecimiento de restos parciales*

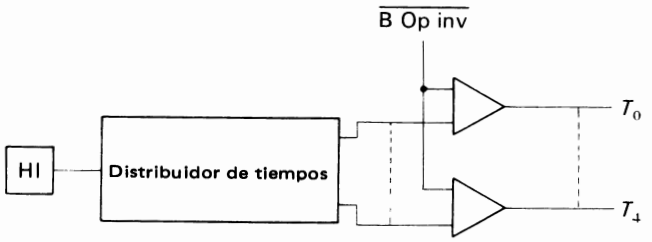


Fig. 34. *Distribuidor de tiempos del divisor por el método de restauración de los restos parciales*

6.2 División por el método de comparación del signo del resto parcial y del divisor

Este método es muy utilizado. Está basado en la comparación del signo de cada resto parcial con el del divisor, en el capítulo 3 del tomo I fue estudiado con detalle. Recordemos las características esenciales de este método:

- Los números negativos son representados por sus complementos a 2.
- El dividendo, al principio de la operación, es contenido en el acumulador y el registro Q, el divisor es introducido en el registro R.

Primeramente se verifica que el divisor no es nulo y que el dividendo no es demasiado negativo. Esta última condición es detectada, cuando los dos primeros dígitos del acumulador $A_1 A_2 = 10$. Si el divisor es positivo, los dividendos muy grandes positivamente serán detectados por la condición $A_1 A_2 R_1 = 010$. La primera operación a efectuar después del desplazamiento a la izquierda del acumulador y del registro Q, es una sustracción o una adición del divisor sobre el contenido del acumulador, según que el signo del dividendo sea igual u opuesto al del divisor.

La existencia de un arrastre más allá del signo del acumulador, a condición de que su contenido no sea nulo, indica un rebasamiento de capacidad del resultado final *cuando el divisor es negativo,* que es detectado de este modo desde la primera operación. Las tres condiciones de invalidez de la división son: divisor nulo, dividendo demasiado negativo (*), rebasamiento de capacidad cuando el divisor es negativo, estas tres condiciones son memorizadas en una báscula Overflow "B Overflow" (1 ⇒ B Overflow) que, según el tipo de calculador, detiene el ciclo de ejecución y envía al ciclo I, o elimina las micro-operaciones siguientes del ciclo E hasta el fin de la ejecución. La elección entre estos dos métodos depende del número y de la complejidad de las operaciones a tratar. Si no existe arrastre más allá del signo después de la primera operación de adición o sustracción, se compara de nuevo el signo del primer resto parcial con el del divisor. Si son idénticos, el

(*) O positivamente.

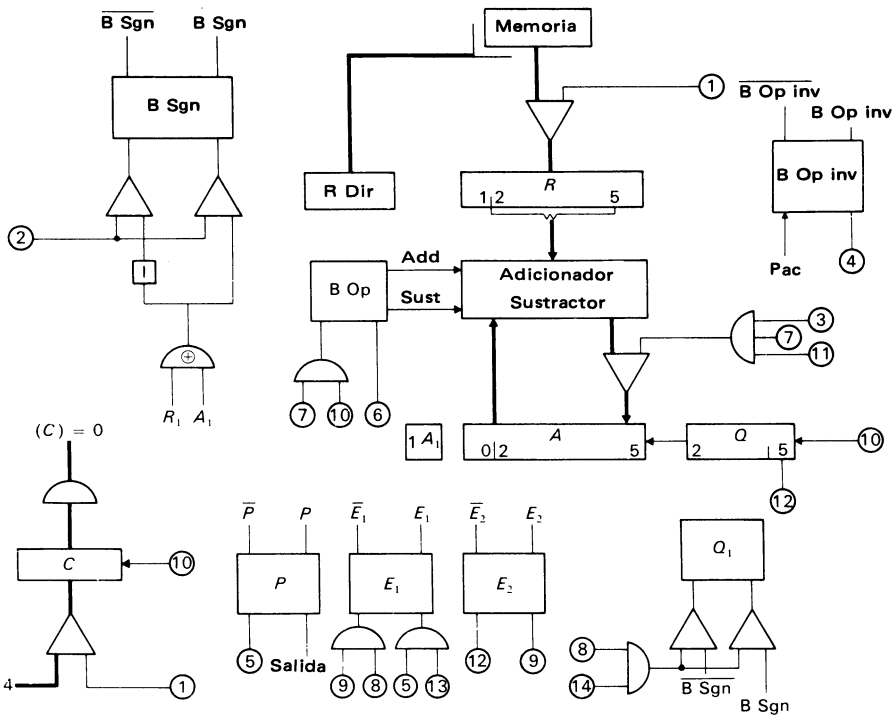


Fig. 35. *Divisor por el método del restablecimiento de restos*

primer dígito del cociente es 1 ($1 \Rightarrow Q_5$) y la operación siguiente, después del desplazamiento de una posición a la izquierda del conjunto acumulador-registro Q, es una sutracción del divisor del contenido del acumulador. Si son opuestos el primer dígito del cociente es 0 y la operación siguiente, después del desplazamiento a la izquierda, es una adición del divisor al contenido del acumulador. A cada desplazamiento disminuye en una unidad el contenido del contador de desplazamientos C, en el que se ha introducido al principio del ciclo de preparación el número de desplazamientos necesarios para efectuar la división más una unidad (*). La adición y la sustracción son ordenadas por una báscula operación B Op ($0 \Rightarrow$ B Op para la sustracción, $1 \Rightarrow$ B Op para la adición) que memoriza la operación a efectuar después del desplazamiento, al mismo tiempo que se introduce el primer dígito del cociente en Q_5. Se continúa de este modo hasta que el contenido del contador de desplazamientos sea igual a 1. En este momento el acumulador contiene el último resto y el registro Q el cociente. El último resto deberá sufrir una corrección si su signo es opuesto al dividendo. En este caso se le añade o se le quita el divisor sin desplazamiento previo, según que

(*) Depende de la realización.

su signo sea opuesto o idéntico al del divisor. El contenido del contador de desplazamiento habrá sido previamente puesto a cero. El cociente deberá igualmente sufrir una corrección en los casos siguientes:

- Dividendo positivo y divisor negativo.
- Dividendo negativo y divisor positivo salvo en el caso en que el resto sea nulo.
- Dividendo y divisor negativos y resto nulo.

En todos estos casos la corrección consiste en añadir 1 al cociente. Hemos estudiado en detalle todos estos casos en el capítulo 3 del tomo I, con ejemplos prácticos. El lector puede referirse a ellos. Finalmente queda una última verificación a efectuar si el cociente no es nulo, que consiste en comparar el signo del resultado con el que ha sido elaborado al principio del ciclo de preparación y memorizado en la báscula "B Sgn". Si los signos son iguales, el resultado es válido. Si son opuestos, el resultado es inválido. Entonces nos encontramos en presencia de un rebasamiento de capacidad del registro Q cuando el divisor es positivo. Las figuras 37, 38 y 39 representan el desarrollo del ciclo de preparación y de ejecución de la división. La figura 36 representa el esquema del distribuidor de tiempos. Se puede observar que son necesarios siete tiempos elementales T_0 a T_6 para realizar el ciclo de preparación y de ejecución de la división, y que ésta solamente podrá efectuarse a condición de que la báscula B Overflow permanezca en 0 durante toda la operación.

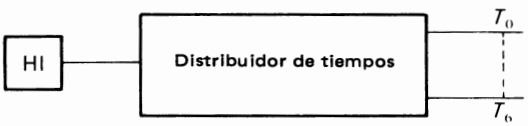


Fig. 36. *Distribuidor de tiempos para la división por el método de comparación de los signos de los restos parciales y del divisor*

Los test de invalidez son efectuados durante el ciclo de preparación P excepto el último efectuado al final del ciclo de ejecución. El ciclo de ejecución está a su vez dividido en dos ciclos, E_1 y E_2, uno para la división, otro para la corrección del resto y del cociente.

Se puede observar (fig. 39) que la corrección del cociente necesita la transferencia simultánea del cociente al acumulador y del resto al registro Q. Este último está protegido ($A \Leftrightarrow Q$). Las figuras 40 y 41 representan los esquemas de realización de los ciclos de preparación y de ejecución. Las figuras 42 y 43 representan las condiciones de corrección del resto y del cociente. Se corregirá el resto cuando la báscula B Sgn esté en 0 y que el signo del resto sea opuesto al del divisor, o cuando la báscula B Sgn esté en 1 y que el signo del resto sea idéntico al del divisor. El resto se corregirá pues, si no es nulo, cuando su signo sea opuesto al signo del dividendo como vimos anteriormente. Igualmente se corregirá el cociente cuando el resto sea nulo y el divisor negativo, o cuando el resto no sea nulo y cuando la báscula B Sgn = 1 (signos del dividendo y del divisor opuestos). La figura 44 representa el esquema general del divisor. En cuanto la báscula B Overflow pasa a 1 elimina las micro-operaciones. El distribuidor de tiempos sigue funcionando pero las micro-operaciones no son ejecutadas.

Nota. Una báscula B Resto memoriza durante la división un resto igualmente nulo. Si al final de la división el último resto no es nulo, éste volverá a ponerse en cero. Esta báscula B Resto es puesta en 0 cada vez que se detecta un 1 en los pesos restantes del dividendo. Esto se obtiene examinando los pesos Q_1 del registro Q a cada impulso de avance. Si éste es 1 pondrá a cero la báscula B Resto indicando un resto no nulo. Si los pesos restantes del dividendo son 0, la báscula B Resto permanecerá en 1 durante toda la división, indicando un resto final nulo. En ciertos casos éste será diferente de 0. Verifique el lector que la división de – 144 por + 12 da un resto no nulo, lo que implica una corrección del cociente. Se obtendrá entonces un resultado falso. Es necesario poner a cero el resto al final de la división siendo entonces el cociente exacto. Este ejemplo es estudiado en el anexo. El lector puede remitirse a él.

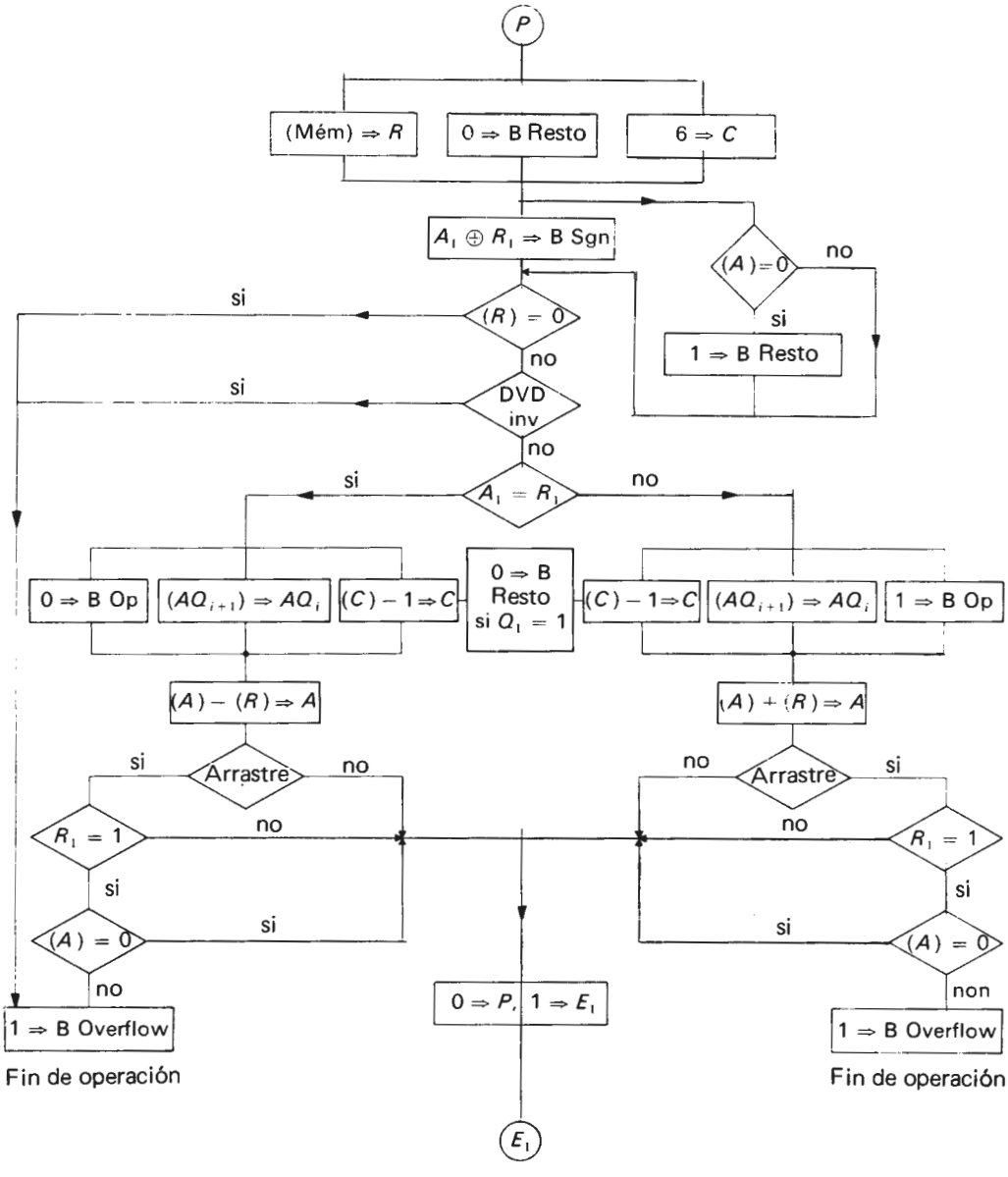


Fig. 37. *Ciclo de preparación de la división por el método de comparación del signo de los restos parciales y con el signo del divisor. DVD inv: Dividendos inválidos*

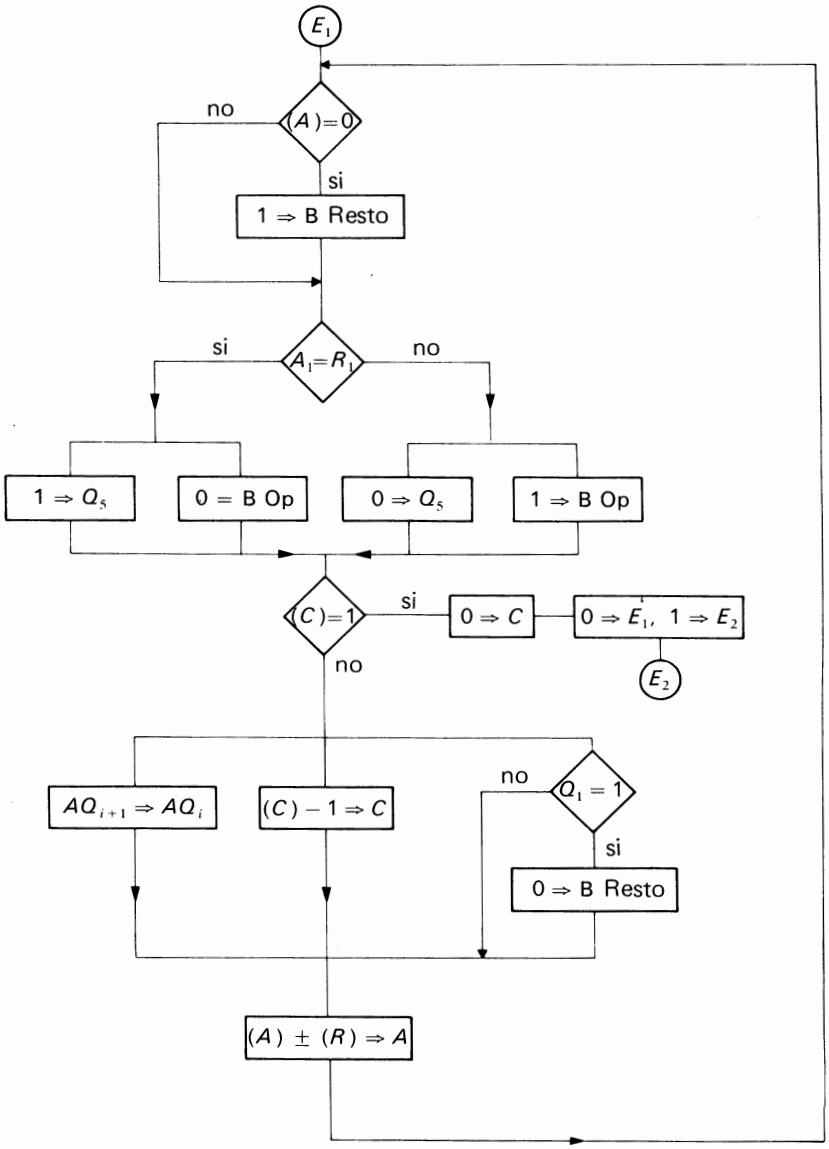


Fig. 38. *Ciclo de ejecución (primera parte) de la división por el método de comparación del signo de los restos parciales con el signo del divisor*

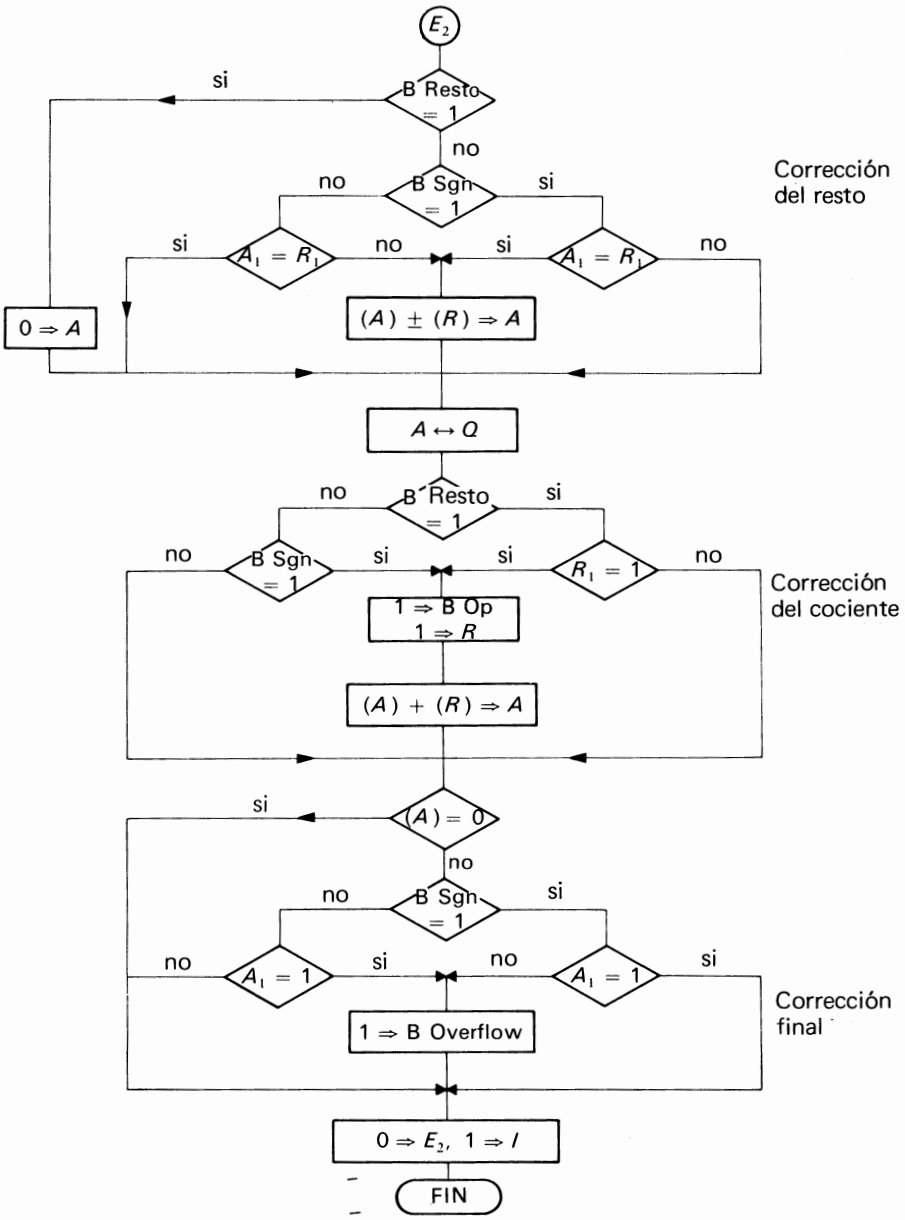


Fig. 39. *Ciclo de ejecución (segunda parte) de la división por el método de comparación del signo de los restos parciales con el signo del divisor*

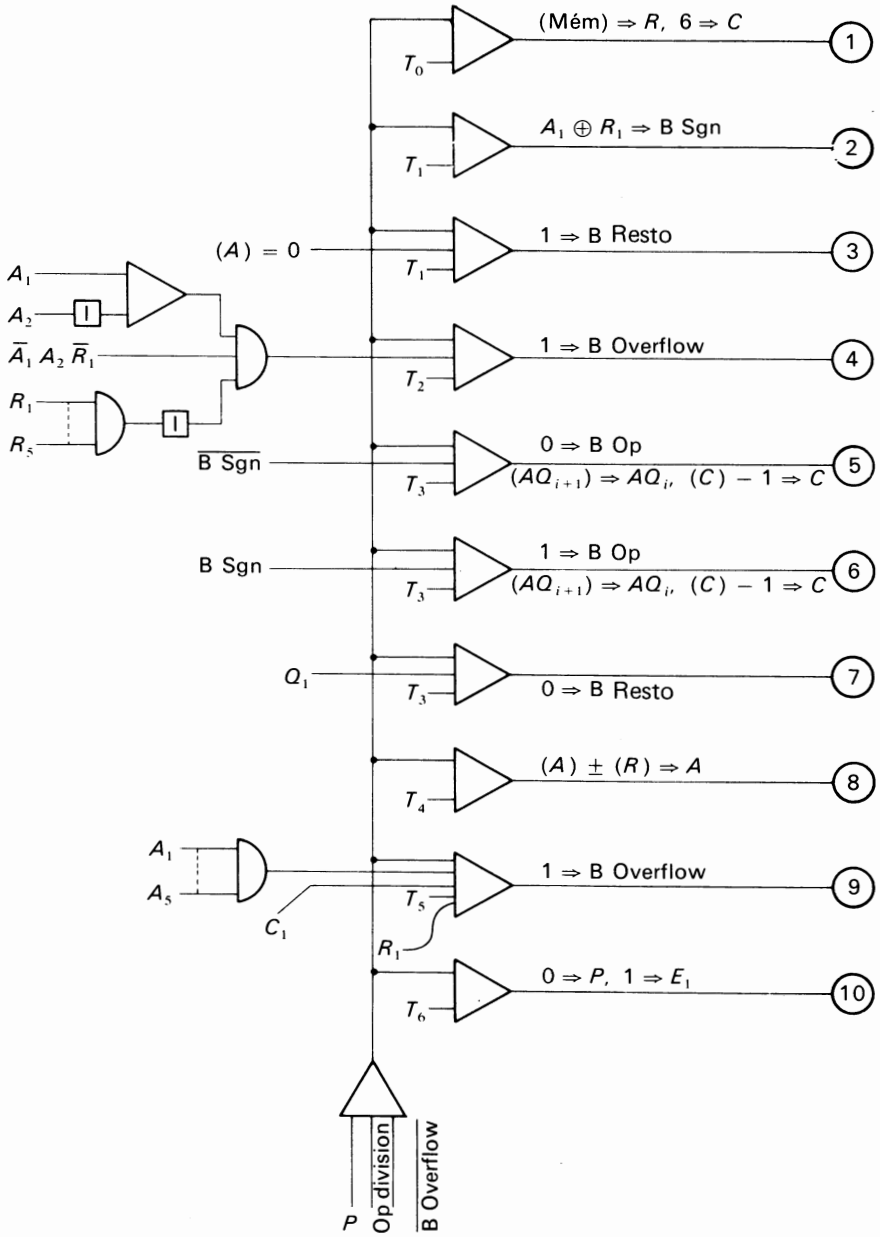


Fig. 40. *Realización del ciclo de preparación de la división por el método de comparación del signo de los restos parciales con el signo del divisor*

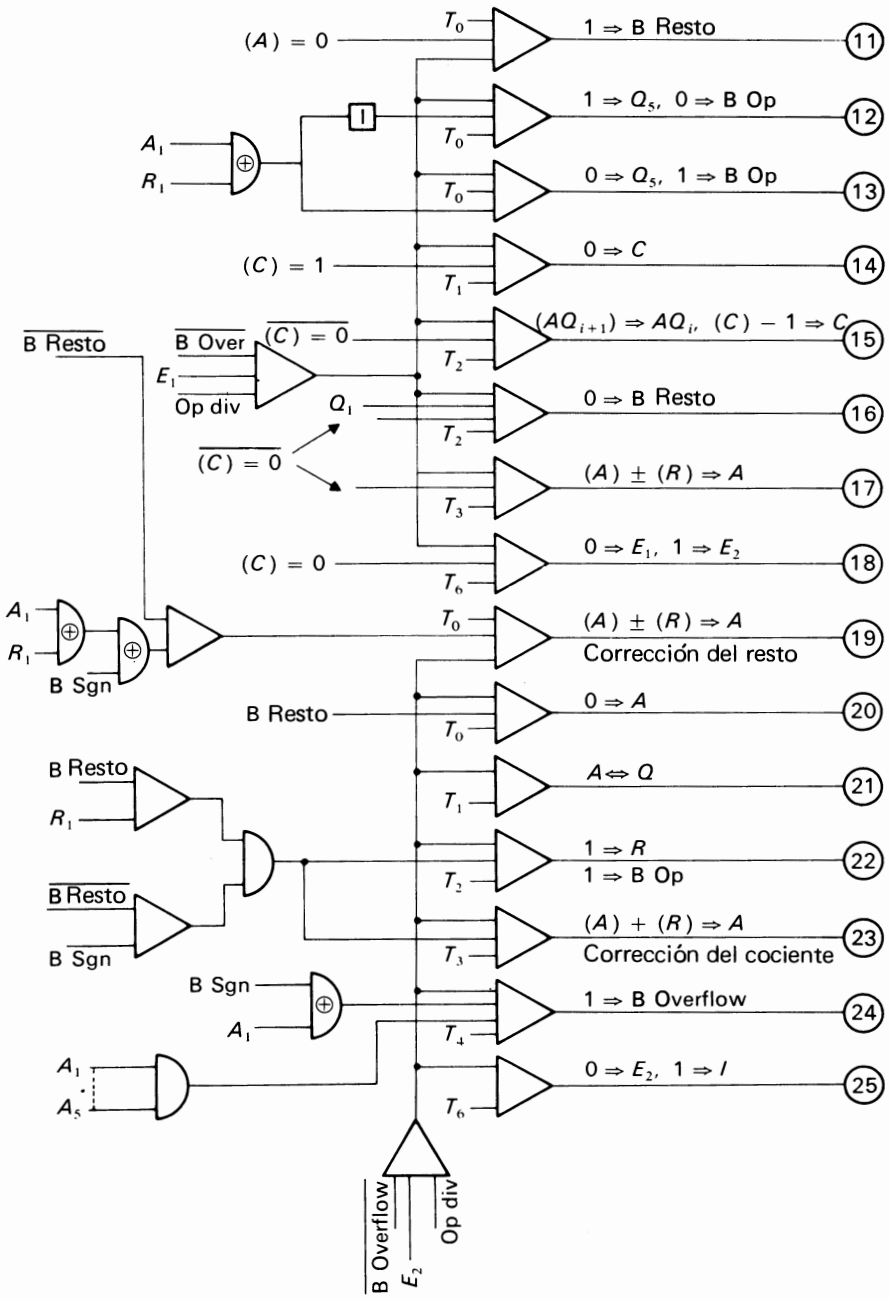


Fig. 41. *Realización del ciclo de ejecución de la división por el método de comparación del signo de los restos parciales con el signo del divisor*

Signo del dividendo	Signo del divisor	Signo del resultado (B Sgn)	Signo del resto	Corrección del resto
0	0	0	0	no
0	⓪	⓪	①	si
0	1	1	0	no
0	[1]	[1]	[1]	[si]
1	[0]	[1]	[0]	[si]
1	0	1	1	no
1	①	⓪	⓪	si
1	1	0	1	no

Fig. 42. *Condiciones de corrección del resto*

Signo del dividendo	Signo del divisor	Signo del resultado (B Sgn)	Resto Nulo	Corrección del cociente
0	0	0	si	no
0	0	0	no	no
0	①	1	si	si
0	1	[1]	[no]	[si]
1	0	1	si	no
1	0	[1]	[no]	[si]
1	①	0	si	si
1	1	0	no	no

Fig. 43. *Condiciones de corrección del cociente*

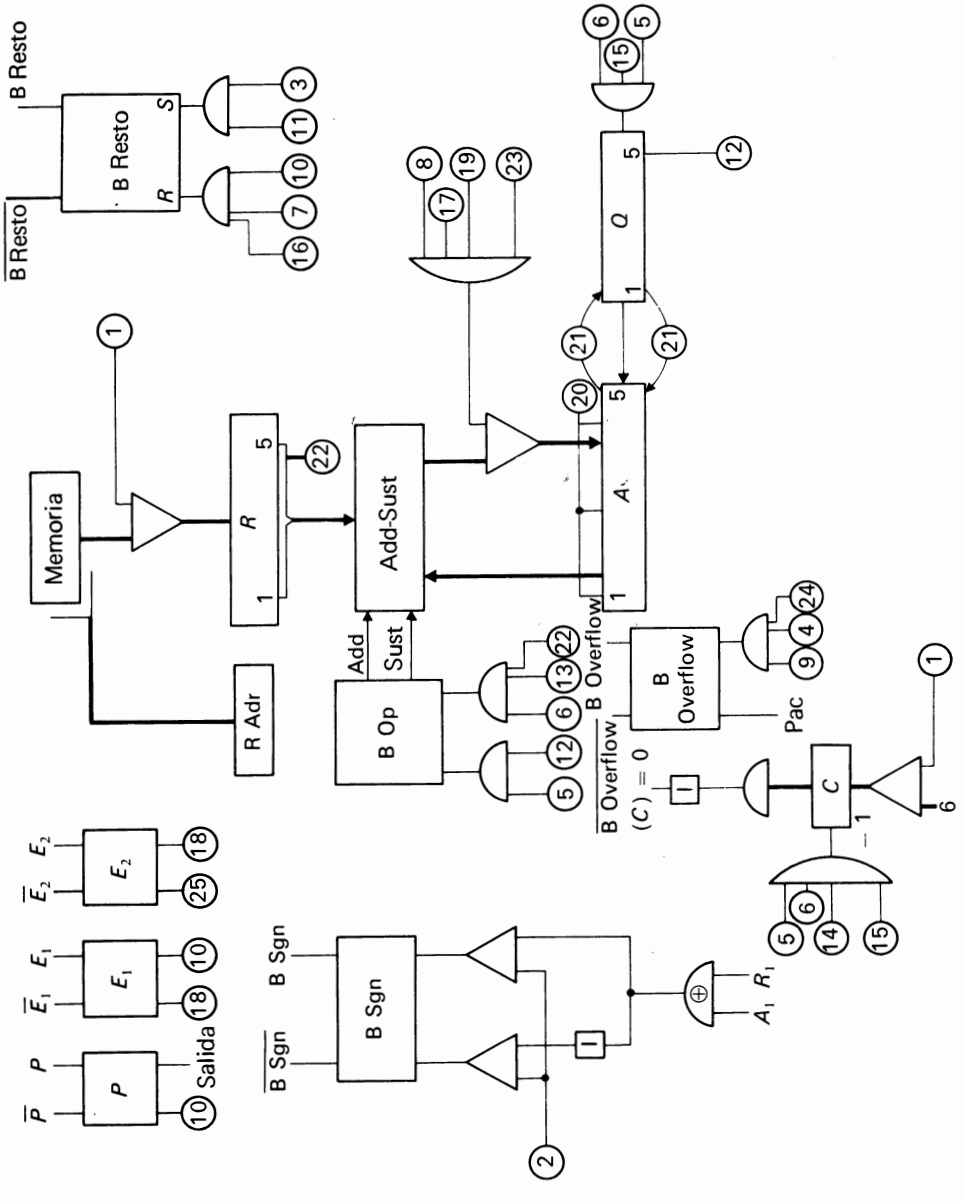


Fig. 44. *Divisor según el método de comparación del signo de los restos parciales y del divisor*

7. Operaciones lógicas «Y» y «O exclusivo»

Hemos visto, en el capítulo 1, que las operaciones lógicas "Y" y "O exclusivo" pueden ser realizadas igualmente por el adicionador-sustractor. Estas operaciones también forman parte de las operaciones realizables por la unidad aritmética y lógica. La figura 15 del capítulo 2 da un ejemplo de un adicionador-sustractor que realiza igualmente las operaciones "Y" y "O exclusivo". La realización de estas operaciones solamente necesita dos tiempos elementales para la preparación y la ejecución. Las figuras 45 y 46 representan los diagramas y la realización de los ciclos de preparación y de ejecución de estas dos operaciones. La figura 47 representa el esquema general del operador que realiza las dos funciones "Y" y "O exclusivo".

7.1 Observación importante

En el caso práctico en el que el adicionador-sustractor ejecuta todas las operaciones estudiadas anteriormente, la que contendrá mayor número de micro-operaciones (en general la división), impondrá la elección del distribuidor de tiempos. De este modo, si se necesitan $n + 1$ micro-operaciones para realizar la división, el distribuidor de tiempos contendrá $n + 1$ tiempos elementales de T_0 a T_n. La preparación y la ejecución de las demás operaciones a excepción de la división contendrán igualmente $n + 1$ tiempos elementales, pero algunos de estos tiempos no serán utilizados. Para evitar este inconveniente el ciclo tendrá una o varias básculas E según que una de las dos operaciones más complicadas (multiplicación o división) necesiten varios ciclos de ejecución. En resumen, es la operación más complicada la que impone la distribución de los tiempos (timing) de todas las operaciones así como una parte de los circuitos del hardware.

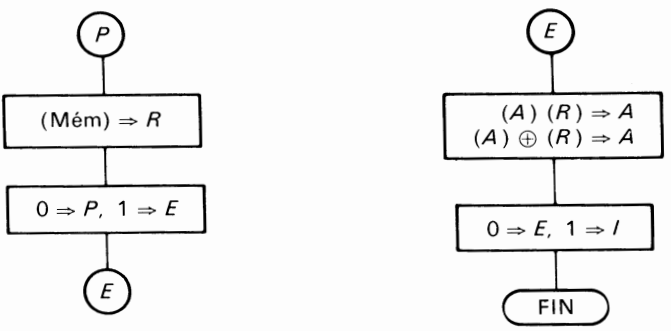


Fig. 45 *Ciclos de preparación y de ejecución de las operaciones lógicas "Y" y "O exclusivo"*

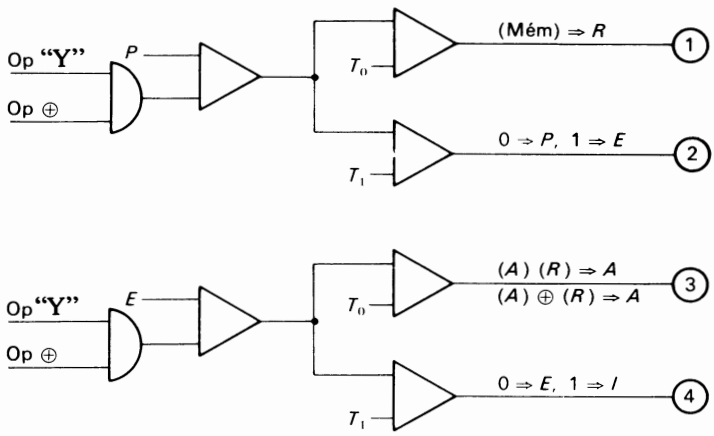


Fig. 46. *Realización de los ciclos de preparación y de ejecución de las operaciones "Y" y ⊕*

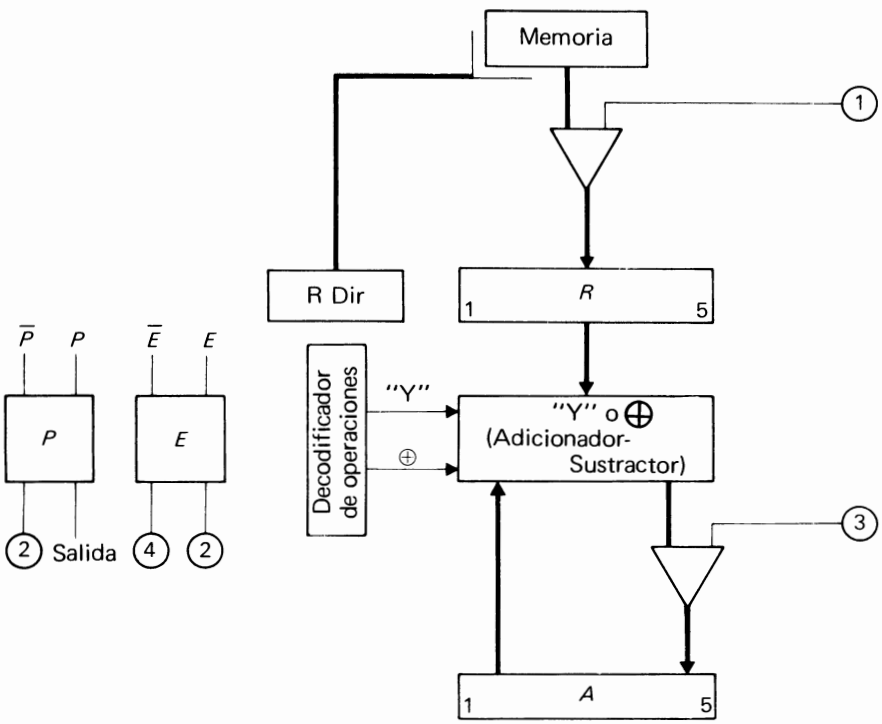


Fig. 47. *Esquema general de las operaciones lógicas "Y" y "O exclusivo"*

8. Las máquinas en serie

En las máquinas en serie el tratamiento de la información se efectúa peso por peso, a diferencia del caso de las máquinas en paralelo, en las que los cálculos se efectúan simultáneamente sobre todos los pesos. Se trata pues de máquinas más lentas, pero más simples desde el punto de vista complejidad de circuitos. La unidad de tiempo no es un impulso elemental del reloj, sino generalmente, un número de impulsos de reloj igual al número de dígitos de una palabra de memoria. Los algoritmos de las operaciones aritméticas son los mismos estudiados anteriormente. La concepción general de la unidad airtmética es también la misma. La única diferencia importante reside en la realización del adicionador que trata solamente un peso a la vez, la suma o la diferencia de dos números binarios se efectúa peso por peso. El lector puede referirse al esquema del adicionador-sustractor en serie estudiado en el capítulo 2. Recordemos que las máquinas en serie, no obstante, son menos utilizadas que las máquinas en paralelo debido al hecho del tiempo de tratamiento más largo. Sin embargo, algunas máquinas funcionan de forma mixta (paralelo-serie), es interesante estudiar cómo se realizan las operaciones aritméticas en este modo de funcionamiento.

8.1 Adición y sustracción

8.1.1 *Números representados en valor absoluto y con signo*

Supongamos que el utilizador dispone de los circuitos de sincronización de las diferentes operaciones. Solamente nos interesaremos por la ejecución de las operaciones. Igualmente se supone que la carga del registro R se efectúa en serie. Un esquema *muy* general del adicionador-sustractor está representado en la figura 48. Este efectúa las operaciones $(R) \pm (A) \Rightarrow A$. Se supone que el registro A contiene previamente números a adicionar. El adicionador-sustractor corresponde al estudiado en el capítulo 2, figura 29. El algoritmo de la adición y de la sustracción es el mismo que se utiliza en las máquinas en paralelo. Una vez correctamente cuadrados los dos números, en sus registros respectivos, éstos están sometidos al avance simultáneamente. Desde que aparecen los dígitos de menor orden en A_5 y R_5, el adicionador-sustractor efectúa, según el código de operación, la adición o la sustracción de estos dos pesos y hace aparecer sobre sus salidas la suma o la diferencia X_i y un arrastre eventual C_i. Cuando aparece el impulso de avance siguiente X_i es introducido en el peso de mayor orden del acumulador (A_2) al mismo tiempo que éste desplaza su contenido de una posición a la derecha. El peso contenido en A_5 se pierde. El arrastre eventual C_i es tomado en cuenta por la

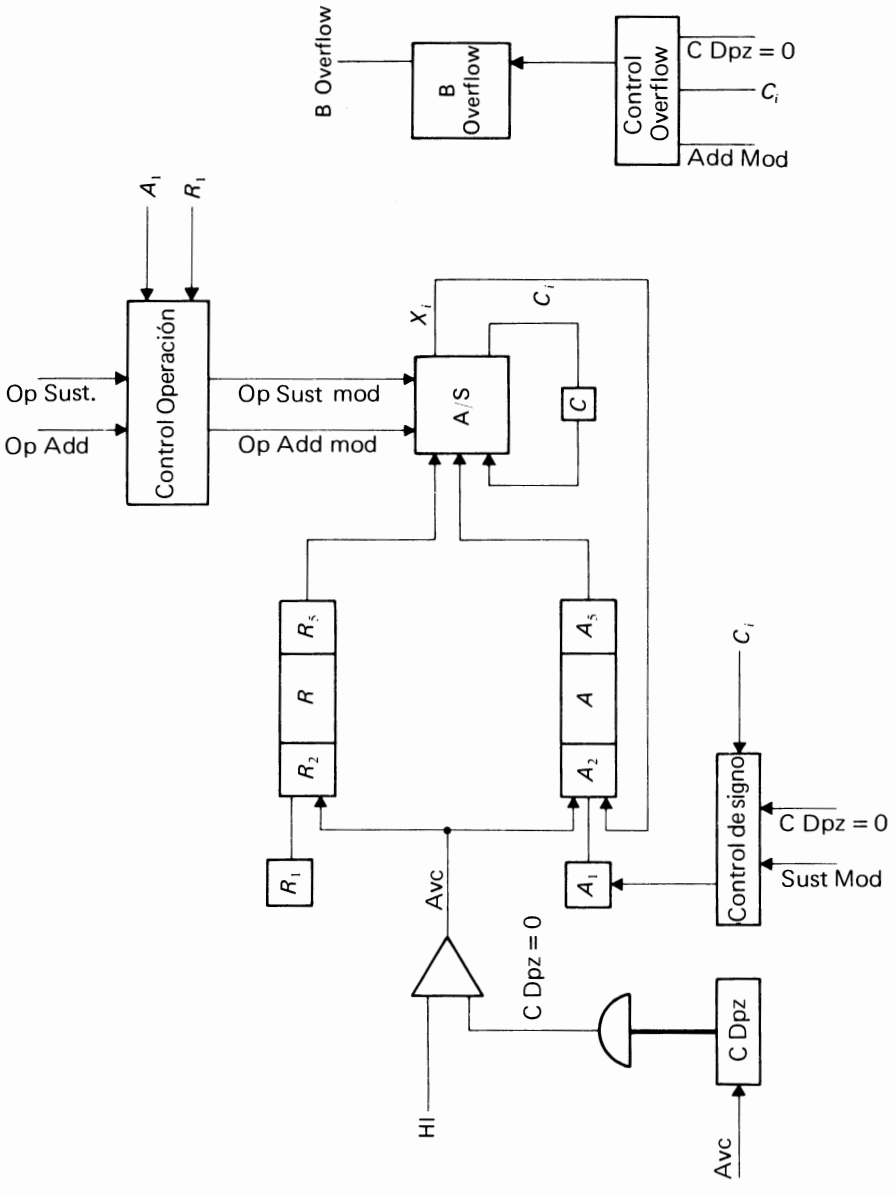


Fig. 48. *Adicionador-sustractor en serie. Números expresados en valor absoluto y con signo*

báscula C que es introducida en la entrada del adicionador-sustractor. Al mismo tiempo el registro R desplaza igualmente su contenido de una posición a la derecha. El contenido de R_5 se pierde y éste contiene entonces el segundo peso de menor orden (R_4). El adicionador-sustractor procede entonces como anteriormente efectuando la suma o la diferencia de los segundos pesos de menor orden, y así para todos los demás pesos. Una vez efectuada la última operación sobre el contenido de A_2 y R_2, se bloquea el avance sobre los dos registros. Si la báscula C contiene un arrastre y efectuamos una sustracción de los módulos, el resultado del registro A aparece en sus complementos a 2. Para obtener un resultado en valor absoluto y con signo, se pone a cero el registro R y se efectúa $(A) - (R) \Rightarrow A$. Finalmente se cambiará el signo de A_1. Si existe un arrastre más allá del peso de mayor orden, efectuando una adición de los módulos, una báscula B Overflow será puesta en 1 e indicará un resultado inválido. Un contador de desplazamientos C dpl asegura el control del avance de los registros.

El esquema general así como todos los que se expondrán a continuación no hacen intervenir el tiempo, en contraste con el caso de las máquinas en paralelo estudiadas anteriormente, esta supresión tiene como objeto la simplificación de los circuitos.

8.1.2 *Números negativos representados por sus complementos a* 2

Se utiliza el mismo algoritmo que para las máquinas en paralelo. En este caso los números negativos son representados por sus complementos a 2. El signo de cada número está colocado al principio de cada palabra en el peso de mayor orden. Los registros R y A contienen, como en el caso anterior, los números a adicionar o sustraer. Los signos están colocados en A_1 y R_1 y son tratados como si fueran dígitos ordinarios. A cada impulso de avance el adicionador-sustractor efectúa la suma peso por peso de dos números tomando en cuenta el eventual arrastre de C_1 que también añade, al impulso de avance siguiente, a los pesos de orden más elevado de los dos números. Al final de la operación la suma o la diferencia aparece en el registro A. Si el contenido de A_1 es 0, el resultado aparece en valor real, si $A_1 = 1$, el resultado aparece en complemento a 2 de cada una de sus cifras. La operación será inválida, si existe un arrastre del módulo en el signo o un arrastre más allá del signo, pero no ambas condiciones. Un registro R Overflow de dos básculas memoriza los arrastres sucesivos del adicionador sustractor en cada impulso de avance. Al final del avance su contenido será 00, 01, 10 ó 11 según que exista o no un arrastre en el transcurso de las dos últimas operaciones. Tendremos una operación inválida, cuando haya un arrastre del módulo en el signo o un arrastre más allá del signo. Un circuito "O exclusivo" realizado con las salidas de las dos básculas del registro R Overflow da la condición de invalidez Overflow. El esquema general del adicionador está representado en la figura 19. El avance, como anteriormente, está controlado por un contador de desplazamientos.

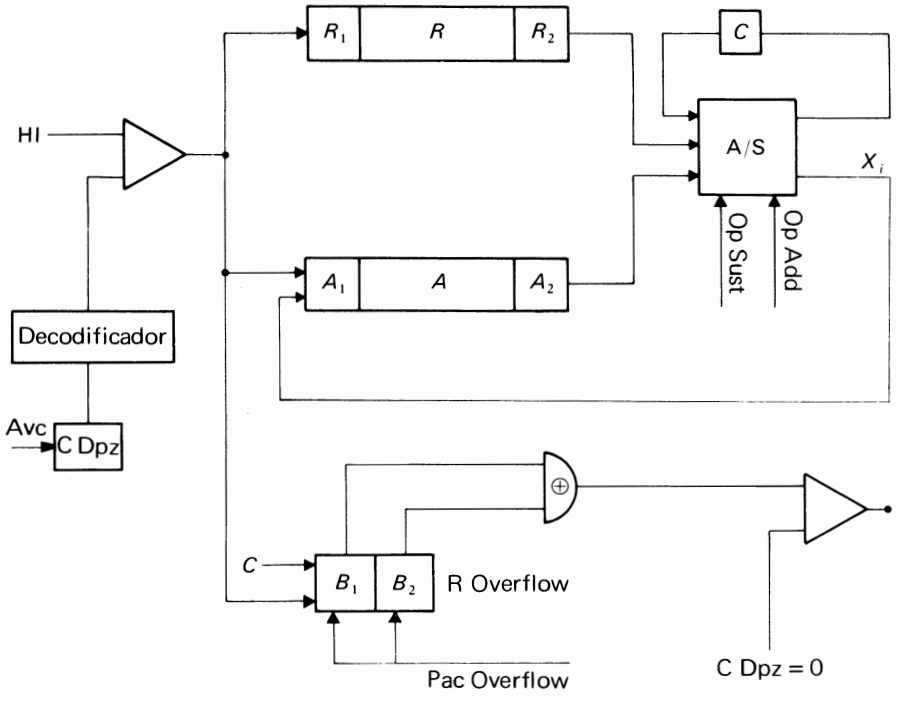


Fig. 49. *Adicionador-sustractor en serie. Números negativos representados por sus complementos a2*

9. Multiplicación en serie

9.1 Números representados en valor absoluto y con signo

El algoritmo es idéntico al de las máquinas en paralelo. La unidad aritmética también contiene los registros R, A y Q. El multiplicando está en R, el multiplicador en Q, al principio de la operación. Los tres registros están sometidos al avance y pueden desplazar sus contenidos de una posición a la derecha. Los pesos A_1, R_1 y Q_1 que contienen los signos de los operandos y el signo del resultado no están sometidos al avance. Al principio de la operación el acumulador es puesto a cero. Al final de la operación el resultado aparece en el acumulador y en el registro Q. El signo del resultado está en A_1. Los signos son tratados separadamente. Se examina el último peso Q_5 del registro Q. Si $Q_5 = 1$, se añade el contenido de R al contenido del acumulador $(A) + (R) \Rightarrow A$, apareciendo la primera suma parcial en A. Si $Q_5 = 0$, se añade 0 al contenido del acumulador, lo que equivale a no efectuar ninguna operación. Esta suma se efectúa peso por peso

al ritmo de los impulsos de avance. El adicionador da la suma X_1 y el arrastre C_i de cada peso. Al impulso de avance siguiente X_i y C_i son introducidos respectivamente en A_2 y la báscula C. Después de cuatro impulsos de avance, A contiene la primera suma parcial. Entonces se desplaza el conjunto acumulador-registro Q de una posición a la derecha y el contenido de la báscula de arrastre es introducido en A_2. Esta operación se efectúa bajo el control de un contador de desplazamientos C dpl que disminuye en una unidad a cada cinco impulsos de avance. Según el nuevo valor de Q_5, se añade el contenido de R ó 0 al contenido del acumulador y se desplaza el conjunto acumulador-registro Q de una posición a la derecha. De este modo se continúa hasta que el contenido del contador de desplazamientos sea nulo. Entonces se suprime el avance.

Una última operación consiste en introducir el signo del resultado $R_1 \oplus Q_1 \Rightarrow A_1$. El resultado de la multiplicación aparece en el conjunto AQ. El esquema general del multiplicador en serie está representado en la figura 50.

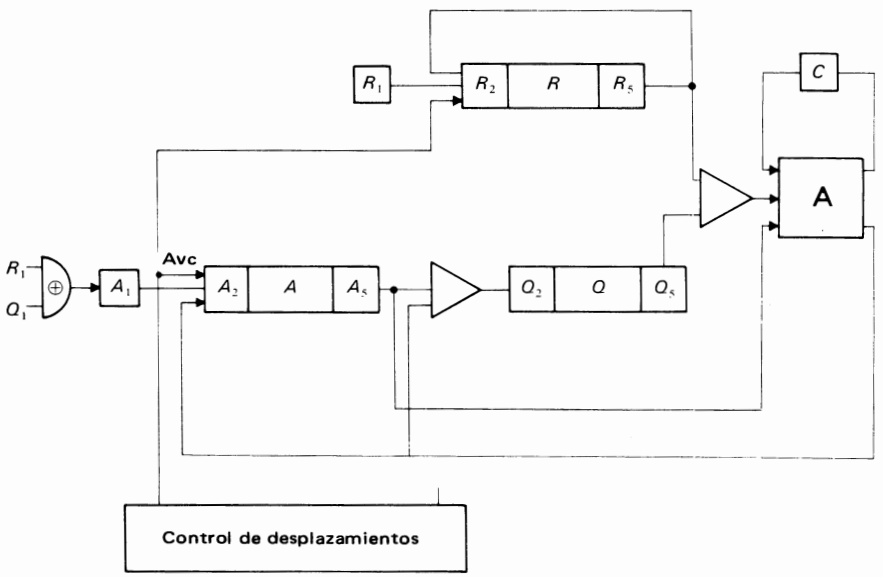


Fig. 50. *Esquema general de un multiplicador en serie para números representados en valor absoluto y con signo*

9.2 Multiplicador en serie que utiliza la representación en complemento a 2 de los números negativos

El algoritmo utilizado es el algoritmo de Booth, segunda versión. El lector puede consultar lo estudiado anteriormente para el multiplicador en paralelo del mismo tipo. El funcionamiento es muy parecido al que hemos visto en el párrafo

precedente. Los signos son tratados como si fueran dígitos ordinarios y el control de las diferentes micro-operaciones es efectuado por el contenido de los dos últimos pesos del multiplicador Q_4 y Q_5. Al principio de la operación el circuito de control de las operaciones es posicionado en adición, seguidamente la sucesión de las operaciones se encadena según la figura 21. El esquema general sin distribuidor de tiempos (timing) está representado en la figura 51.

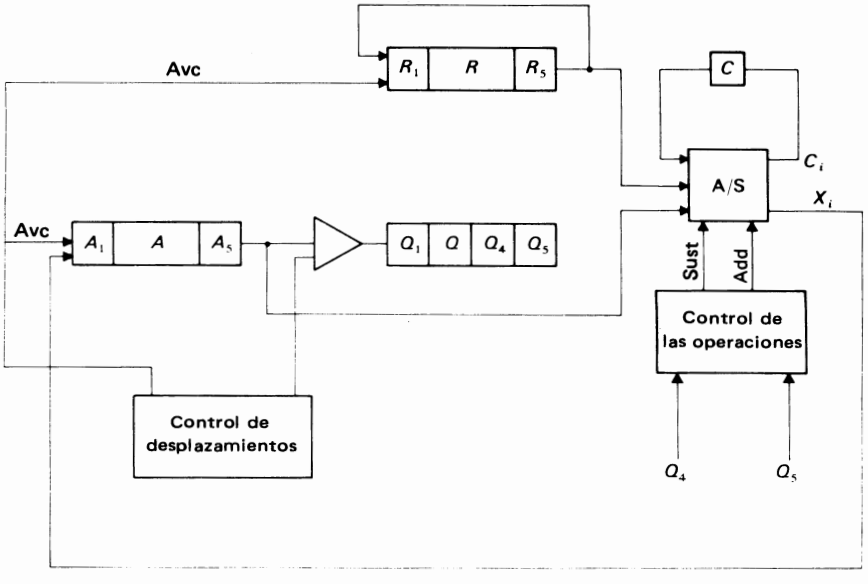


Fig. 51. *Multiplicador en serie que utiliza el algoritmo de Booth (segunda versión)*

10. Divisor en serie

10.1 Números representados en valor absoluto y con signo

El algoritmo utilizado es el mismo que para las máquinas en paralelo. El lector puede referirse a él. El esquema general (sin el tiempo) de un divisor de este tipo está representado en la figura 52. La primera operación es, como siempre, una sustracción del divisor sobre el dividendo. Al principio de la operación el dividendo está en el acumulador y en el registro Q, el divisor en el registro R. El acumulador puede desplazar su contenido de una posición a la derecha a cada impulso de avance. Igualmente el conjunto acumulador-registro Q puede desplazar su contenido de una posición a la izquierda bajo ciertas condiciones. Al principio de la operación el dispositivo de control de las operaciones impone una sustracción al adicionador-sustractor.

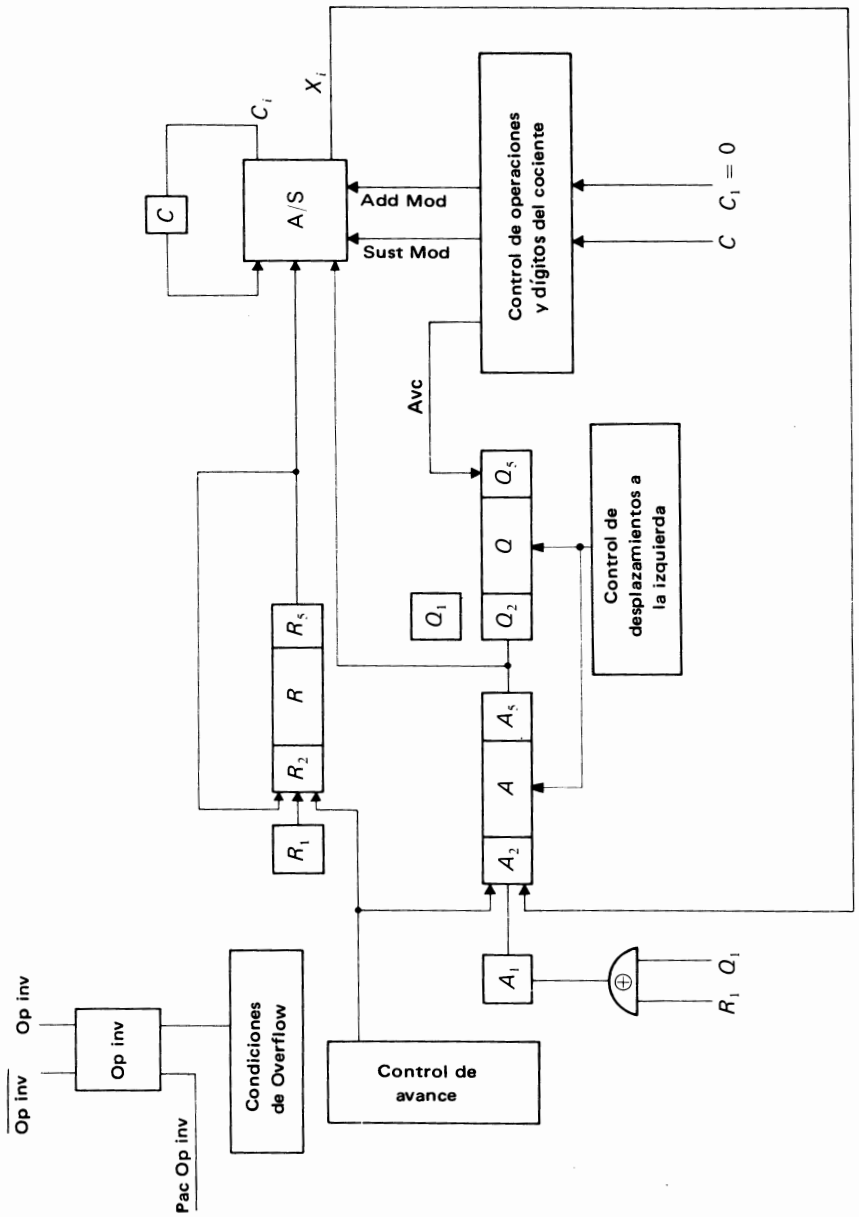


Fig. 52. *Esquema general de un divisor en serie por el método de restauracion de los restos. El tiempo no figura*

Si el signo de la primera sustracción es 0 (Arrastre $C = 0$) la operación será inválida. Para que la operación sea válida es necesario que la primera operación dé un resultado negativo ($C = 1$). En este caso se restablece el dividendo por adición del divisor, y se desplaza seguidamente el conjunto acumulador-registro Q de una posición a la izquierda. Según el signo de cada resto parcial se introducirá en Q_5 un 1 ó un 0. Si el resultado es positivo ($C = 0$) se introduce un 1, y se desplaza de una posición a la izquierda el conjunto AQ. La operación siguiente es una sustracción del divisor. Si el resto es negativo, se introduce un cero, se detiene el desplazamiento a la izquierda y se procede al restablecimiento del resto por adición del divisor. Obsérvese que el signo del resto, antes del desplazamiento, impone la operación siguiente. Para conocer mejor el detalle de las operaciones el lector puede remitirse a lo estudiado para las máquinas en paralelo. La figura 52 representa el esquema general de principio de un divisor de este tipo sin el tiempo.

11. Unidad aritmética en coma flotante

La unidad aritmética que acabamos de estudiar trata los números enteros. Esto supone que la coma que separa la parte entera de los números de su parte decimal está situada inmediatamente detrás del peso de menor orden de los registros. Esta misma unidad aritmética podrá ser utilizada, con algunas modificaciones de detalle para la división, en el caso de una representación fraccionaria de números en coma fija. Siendo los números inferiores a la unidad, la posición de la coma está situada inmediatamente detrás de la posición del signo (2^0). Esto obliga al programador a asegurarse de que la posición de la coma, durante los cálculos, conserva el sitio que le ha sido asignado inmediatamente detrás de la posición del signo. *Para evitar el inconveniente y también para aumentar la capacidad de los registros,* algunos calculadores están equipados de unidades aritméticas que funcionan en *coma flotante.* En este caso la unidad aritmética toma en cuenta automáticamente la posición de la coma. El método consiste en representar a los números en forma exponencial conteniendo dos partes: una *mantisa*, con los dígitos del número a representar, y un *exponente,* expresando la potencia de 2 por la que hay que multiplicar la mantisa para obtener el valor exacto del número. El *exponente* debe ser elegido de tal modo que la mantisa aparezca como un número inferior a 1, con el primer dígito significativo situado inmediatamente detrás de la posición del signo. Se utilizará la representación en valor absoluto y con signo para la mantisa. Algunas máquinas utilizan la representación por complemento a 2 para las mantisas negativas. La representación de un número con signo es de la forma siguiente (fig. 53a):

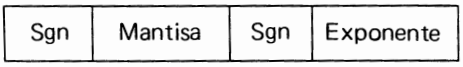


Fig. 53a. *Representación de un número en coma flotante con una posición signo para el exponente*

Algunas máquinas incluyen la posición del signo del exponente, en el exponente mismo.

Ejemplo. Sea un exponente constituido por 6 posiciones binarias más un signo. De este modo se representan los exponentes comprendidos entre los límites:

$$63 \geqslant e \geqslant -64 .$$

Si la posición del signo está incluida en el módulo, el exponente podrá ser representado por los valores binarios comprendidos entre

$$127 \geqslant e \geqslant 0 ,$$

lo que equivale a representar los límites del exponente por las configuraciones siguientes:

0 0 0 0 0 0 0 (– 64)
0 1 1 1 1 1 1 (– 1)
1 0 0 0 0 0 0 0
1 1 1 1 1 1 1 + 63

La representación de un número en coma flotante será pues la de la figura 53b.

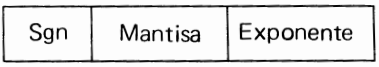


Fig. 53b. *Representación de un número en coma flotante sin posición para el signo del exponente*

La unidad aritmética que trata los números representados en coma flotante es naturalmente más compleja que una unidad clásica ya que es necesario realizar operaciones sobre las mantisas y los exponentes.

Sobre todo será necesario prever una unidad aritmética separada para tratar los exponentes con la configuración adoptada. Esta representación obliga a realizar correcciones en el caso de la adición o la sustracción de los exponentes que consiste, si no hay rebasamiento de capacidad por los valores positivos y negativos en invertir el peso de mayor orden en el resultado con el fin de obtener el mismo de acuerdo con la representación adoptada.

Ejemplo 1. Adicionar los exponentes (+ 1) *y* (− 2) :

	1 0 0 0 0 0 1	(+ 1)
+	0 1 1 1 1 1 0	(− 2)
	1 1 1 1 1 1 1	
Inversión del peso de mayor orden	0 1 1 1 1 1 1	(− 1)

Ejemplo 2. Adicionar los exponentes (+ 62) *y* (− 1) :

	1 1 1 1 1 1 0	(+ 62)
+	0 1 1 1 1 1 1	(− 1)
Arrastre ignorado	0 1 1 1 1 0 1	
Inversión del peso de mayor orden	1 1 1 1 1 0 1	(+ 61)

Ejemplo 3. Sustraer los exponentes (+ 1) *y* (− 3) :

	1 0 0 0 0 0 1	(+ 1)
−	0 1 1 1 1 0 1	(− 3)
	0 0 0 0 1 0 0	
Inversión del peso de mayor orden	1 0 0 0 1 0 0	(+ 4)

Ejemplo 4. Sustraer los exponentes (− 6) *y* (− 57) :

	0 1 1 1 0 1 0	(− 6)
−	0 0 0 0 1 1 1	(− 57)
	0 1 1 0 0 1 1	
Inversión del peso de mayor orden	1 1 1 0 0 1 1	(+ 51)

Los rebasamientos de capacidad del acumulador durante la suma o la diferencia de los exponentes se obtendrá examinando el peso de mayor orden así como la existencia de un arrastre eventual más allá de este peso. Se presentan dos casos de rebasamiento de capacidad por los valores positivos (Overflow) o negativos (Underflow). El cuadro siguiente resume los diferentes casos.

A_8 A_7	Operación	Rebasamiento de capacidad
0 0	Adición	Underflow
1 1		Overflow
0 1	Sustracción	Overflow
1 0		Underflow

Un rebasamiento de capacidad provoca generalmente una interrupción del programa en curso de ejecución.

Adición y sustracción

La adición y la sustracción de dos números en coma flotante se efectúa comparando previamente los exponentes.

Se añade el menor sobre el mayor desplazando la mantisa de un número de posiciones a la derecha igual a la diferencia de los exponentes. Seguidamente se efectúa la suma o la diferencia de las mantisas. El exponente del resultado es el mayor de los exponentes. A continuación, si es necesario, se normaliza el resultado, desplazando la mantisa un cierto número de posiciones a la izquierda con el fin de obtener un bit significativo a la derecha del signo. Se disminuye el exponente del resultado en el mismo número de unidades. Un arrastre eventual es tomado en cuenta en una posición suplementaria del acumulador.

Multiplicación y división

Para multiplicar dos números en coma flotante se multiplican las mantisas y se suman los exponentes. El resultado se normaliza si es necesario.

Para dividir dos números, se dividen las mantisas y se restan los exponentes. El resultado se normaliza si es necesario.

Operaciones inválidas

Dos tipos de operaciones inválidas son eliminadas: adición o sustracción de dos números de mantisa nula y división por una mantisa nula.

Realización de una unidad aritmética en coma flotante

Esta unidad contendrá una unidad aritmética clásica destinada a las operaciones sobre las mantisas.

Un adicionador-sustractor para las operaciones sobre los exponentes con detección de rebasamiento de capacidad y corrección del resultado. Un dispositivo de comparación de los exponentes que desplaza la mantisa del número más pequeño, hacia la derecha hasta que los exponentes coincidan. Un dispositivo de control de la normalización del resultado.

12. Las máquinas decimales

Las máquinas que acabamos de estudiar son máquinas binarias, es decir que los datos son expresados directamente en código binario puro. No obstante, frecuentemente es interesante tratar los datos directamente en forma decimal. En este caso cada dígito decimal es representado en el código BCD por una tetrada. En estas máquinas, resulta, pues, necesario utilizar adicionadores-sustractores decimales idénticos a los de la figura 33 y 38 del capítulo anterior. Existen las máquinas decimales en paralelo, en paralelo-serie y en serie.

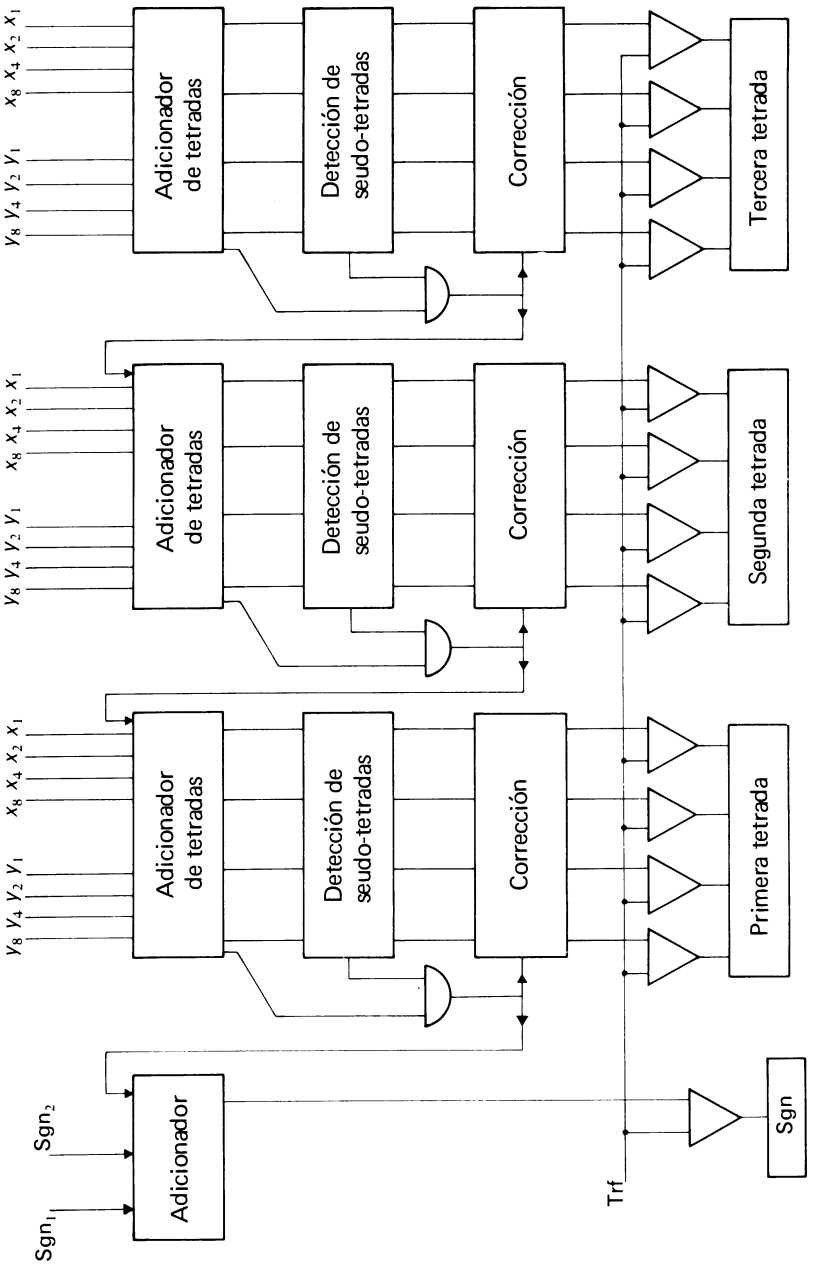


Fig. 54. *Adicionador decimal en paralelo de tres dígitos más el signo*

12.1 Adicionador decimal en paralelo

En un adicionador en paralelo todos los dígitos decimales de los dos números a adicionar son tratados simultáneamente, la detección de una seudo-tetrada o la existencia de un arrastre más allá del orden más elevado, da lugar a una corrección del resultado que depende de la naturaleza del código utilizado. En el caso del código BCD se añade + 110 en la adición y se sustrae + 110 para la sustracción (adición por complemento a 2). Un rebasamiento eventual de capacidad será detectado del mismo modo que en las máquinas binarias. La figura 54 da un ejemplo del adicionador decimal para tres dígitos.

12.2 Adicionador decimal serie-paralelo

El adicionador decimal que acabamos de estudiar es del tipo en paralelo. La adición del conjunto de las tetradas así como la elaboración del signo del resultado es efectuada simultáneamente. Este tipo de adicionador es muy rápido ya que el resultado es obtenido con un impulso de avance (impulso de transferencia de las tetradas al adicionador). Sin embargo, esta solución es muy cara. En efecto, es necesario prever tantos adicionadores de tetradas como pesos haya en los números decimales a adicionar. Un adicionador binario suplementario debe estar previsto para el tratamiento del signo. Si se desea realizar una máquina más rápida será preferible utilizar un adicionador serie-paralelo. En este tipo de adicionador las tetradas a adicionar son introducidas sucesivamente por pares en el adicionador. Este da una suma sin arrastre así como un arrastre que es memorizado en una báscula durante un tiempo igual al que separa dos tetradas. A la llegada de las dos tetradas siguientes, el arrastre es introducido en el adicionador en el menor rango decimal. El adicionador suministra entonces una nueva suma y un arrastre eventual. A la llegada de las segundas tetradas el número que hay en el registro suma (acumulador) sufre un desplazamiento de cuatro dígitos hacia los pesos de menor orden. Este desplazamiento de cuatro pesos puede obtenerse con un solo impulso de avance (desplazamiento de los cuatro pesos, cableado). De este modo se procede para las demás tetradas. *Para simplificar el circuito, este adicionador utiliza el complemento a* 10 *para los números negativos.* Los arrastres más allá del signo son ignorados. El signo puede ser igualmente representado por una tetrada binaria, 0000 para el signo +, 0001 para el signo –. Entonces basta con un solo adicionador de tetrada y de una báscula de arrastre así como un registro suma de 4 n posiciones, siendo n el número de dígitos decimales, comprendido el signo, de los dos números a adicionar. El esquema de la figura 55 representa un adicionador serie-paralelo decimal. Obsérvese que las tetradas llegan en serie y son seguidamente tratadas en paralelo por el adicionador.

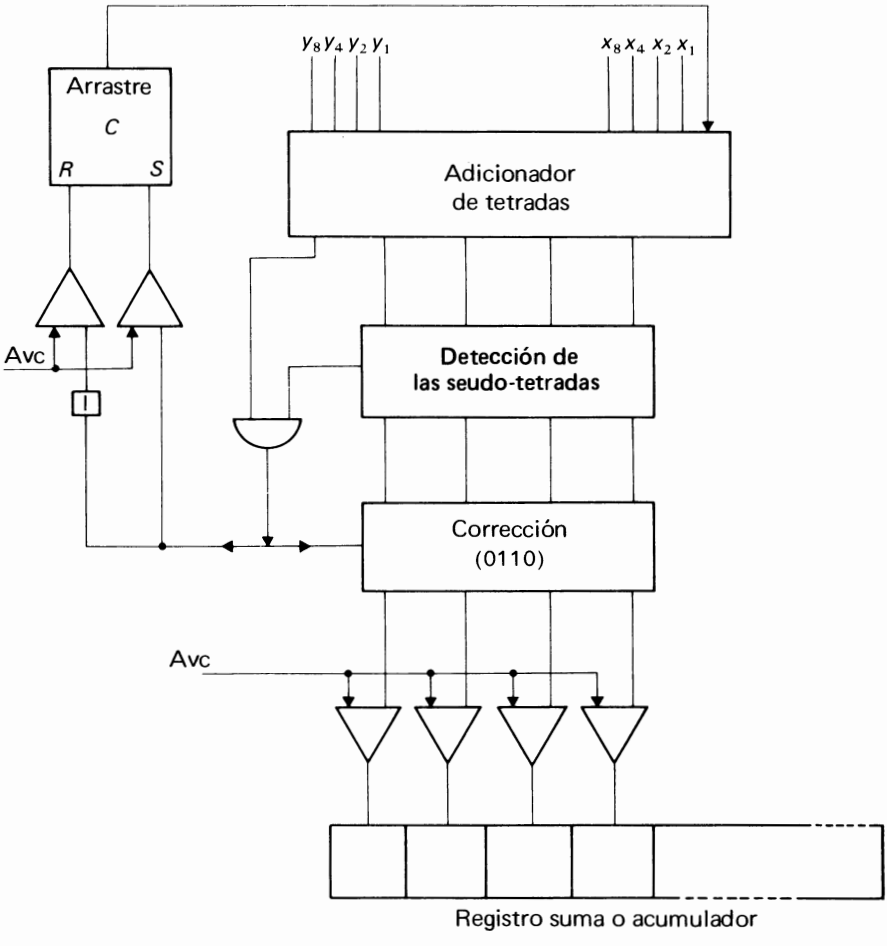


Fig. 55. *Adicionador decimal en serie-paralelo*

Consideremos el caso del ejemplo del capítulo 3, tomo I:

$$(-\ 452) + (136) = -\ 316\ .$$

El equivalente en máquina de -452 es 0001 0101 0100 1000 y el de $+136$: 0000 0001 0011 0110. El signo, contrariamente al caso de la máquina en paralelo, se representa aquí por una tetrada. Los números negativos son complementados a 10. El adicionador efectúa primeramente la suma de las dos primeras tetradas 1000 y 0110 = 1110. Esta suma es identificada como una seudo-tetrada. El circuito de detección engendra un arrastre que es memorizado, coincidiendo con

el impulso de avance, en una báscula de arrastre *C*. El resultado en el registro acumulador es desplazado de cuatro posiciones a la derecha. Seguidamente se transfieren las dos tetradas siguientes 0100 y 0011 al adicionador. Este efectúa la adición de las dos tetradas y del arrastre del orden precedente o sea 1000. Esta suma es reconocida como una tetrada válida y no da lugar a un arrastre. Con el impulso de avance siguiente el registro de arrastre es puesto a cero y la suma transferida al acumulador. El contenido de éste sufre un nuevo desplazamiento de cuatro pesos a la derecha. Se transfieren seguidamente las terceras tetradas 0101 y 0011 al adicionador, el cual da una suma 0110 que representa una tetrada válida. Luego no hay arrastre y la báscula de arrastre permanece en cero. La suma es transferida al acumulador que sufre un nuevo desplazamiento de cuatro pesos hacia la derecha. Las tetradas correspondientes a los signos de los números son transferidas entonces al adicionador. Este da una suma 0001 +0000 = 0001 sin arrastre que es transferida al acumulador. Al final de la adición, el acumulador contiene la expresión siguiente:

0001	0110	1000	0100
–	6	8	4

lo que da después de complementar a 10:

–	3	1	6

Adicionador decimal en serie

Hemos examinado en el capítulo 1, el caso del adicionador decimal en paralelo. El esquema de la figura 56 representa su equivalente en serie. Comprende cuatro registros: *A, B, C y D* así como dos adicionadores en serie. C_1 y C_2 representan las básculas de arrastres asociadas a cada uno de los adicionadores. Un dispositivo de control, desplaza el contenido de los registros y provoca el proceso de corrección, si éste tiene que realizarse. Los dos primeros números representados en código BCD por una posición de tetradas están colocados en los registros *A* y *B* y sumados en el primer adicionador. La báscula C_1 toma en cuenta el arrastre eventual que es adicionado a los dos pesos siguientes de las tetradas.

El resultado de la adición de las dos primeras tetradas aparece en el registro *D*. El circuito de corrección detecta entonces si el resultado es inválido y corrige el resultado.

Recordemos que el resultado será inválido si:

– el contenido del registro *D* es una seudo-tetrada,

– o si existe un arrastre más allá del mayor orden del resultado.

Las seudo-tetradas representan las combinaciones inválidas de 10 a 15. El cuadro siguiente resume los casos de invalidez.

Valor decimal	C_1	S_3^*	S_2^*	S_1^*	S_0^*
10	0	1	0	1	1
11	0	1	0	1	1
12	0	1	1	0	0
13	0	1	1	0	1
14	0	1	1	1	0
15	0	1	1	1	1
16	1	0	0	0	0
17	1	0	0	0	1
18	1	0	0	1	0
19	1	0	0	1	1

El examen de este cuadro pone de relieve que estamos en presencia de un resultado inválido si $C_1 = 1$ o si S_1 o S_2 son iguales a 1 y S_3 igual a 1; o sea:

$$P = C_1 + S_3^*(S_2^* + S_1^*) .$$

Un circuito de detección conectado sobre el registro *D* detecta esta condición y autoriza la corrección. Hemos visto en el tomo I que esta corrección consiste en añadir $6_{10} = 0110$ al resultado y poner en 1 la báscula C_1 (arrastre más allá de la primera tetrada). El factor de corrección 0110 es introducido en el registro *C* y sumado al segundo adicionador. El resultado corregido es introducido en el registro *A* que hace la función de acumulador. Al mismo tiempo la báscula C_2 es puesta a cero. Esta representa un arrastre que si existe debe ser despreciado. En efecto, la adición de $6_{10} = 0110$ representa la sustracción del complemento a 2 de $10_{10} = 1010$ y en este caso se desprecia un arrastre más allá del peso de mayor orden. Al final del tratamiento el resultado de la adición decimal está en *A*, el contenido de *B* se pierde.

12.3 Multiplicación decimal

Existen varios modos de hacer una multiplicación decimal. La más inmediata consiste en añadir sucesivamente el multiplicando a sí mismo tantas veces como indica el dígito de menor orden del multiplicador. Este se encuentra en el registro decimal *Q* que disminuye en una unidad cada vez que el multiplicando se suma a sí mismo. Cuando la última tetrada del multiplicador es nula se detiene el proceso de adición. La primera suma parcial está en el acumulador. Seguidamente se desplaza de una posición decimal el conjunto acumulador-registro *Q*. La tetrada de menor rango decimal contiene ahora el segundo menor peso del multiplicador. Se procede como anteriormente hasta que la última tetrada sea nula. Nuevamente se desplaza el conjunto acumulador-registro *Q* de un rango decimal a la derecha y vuelve a comenzar el proceso.

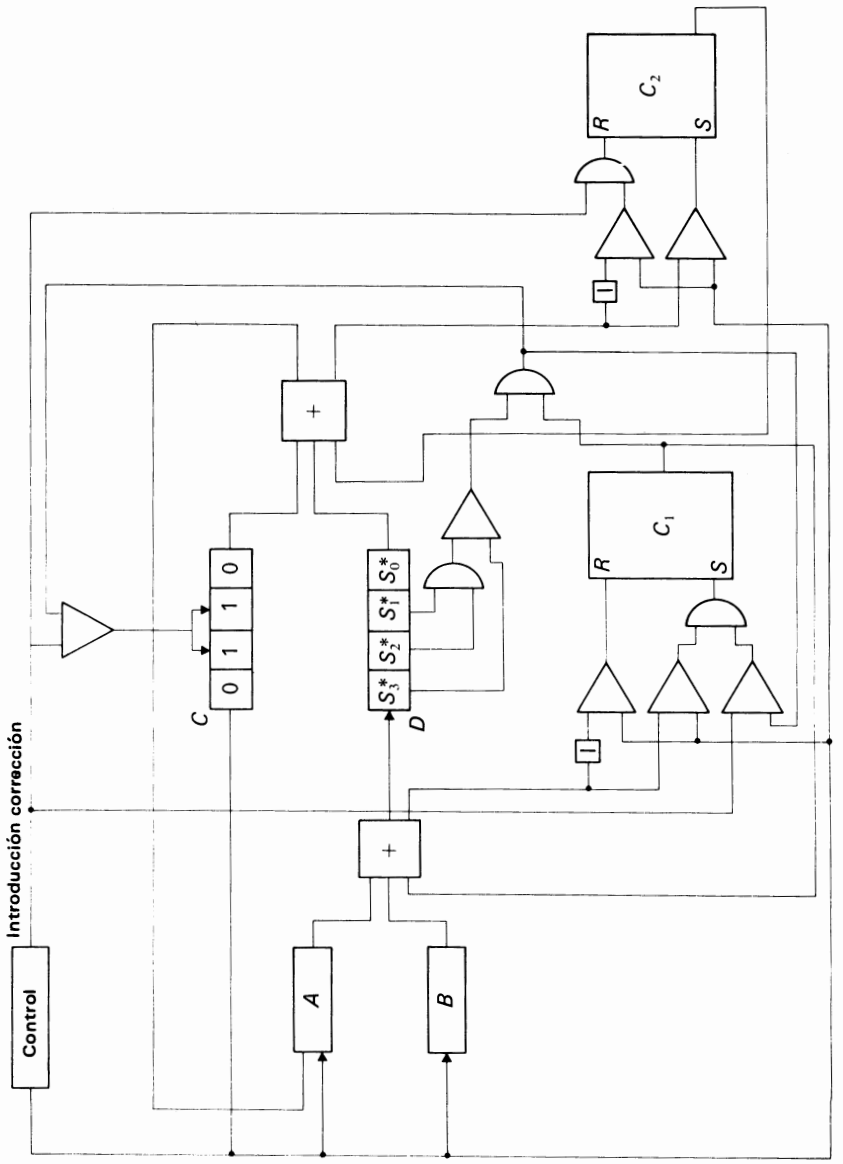


Fig. 56. *Esquema general de un adicionador decimal en serie*

Un contador de desplazamientos contiene al principio de la operación el número de desplazamientos necesarios para efectuar la multiplicación. Cuando el contenido de este contador es nulo la multiplicación ha terminado. Esta configuración es idéntica a la de las máquinas en paralelo. *No obstante, en este caso, cada dígito decimal está representado por una tetrada.* Se puede concebir pues que una máquina de esta naturaleza sea de coste elevado. Otro modo de realizar la multiplicación consiste en formar previamente en la máquina un cierto número de múltiplos del multiplicando. De este modo se aumentará la rapidez de la multiplicación. En general se formarán los productos siguientes, siendo M el multiplicando:

$$P_1 = M\,, \quad P_2 = 2\,M\,, \quad P_5 = 5\,M\,, \quad M_8 = 8\,M\,,$$

los productos intermedios P_3, P_4, P_6, P_7 y P_9 se forman del modo siguiente:

$$P_3 = M + 2\,M\,, \qquad P_7 = 2\,M + 5\,M\,,$$
$$P_4 = 2\,M + 2\,M\,, \qquad P_9 = M + 8\,M\,.$$
$$P_6 = M + 5\,M\,,$$

Tetradas decimales x_8, x_4, x_2, x_1	C_i	Tetradas dobles X_8, X_4, X_2, X_1
0 0 0 0	0	0 0 0 0
0 0 0 1	0	0 0 1 0
0 0 1 0	0	0 1 0 0
0 0 1 1	0	0 1 1 0
0 1 0 0	0	1 0 0 0
0 1 0 1	1	0 0 0 0
0 1 1 0	1	0 0 1 0
0 1 1 1	1	0 1 0 0
1 0 0 0	1	0 1 1 0
1 0 0 1	1	1 0 0 0
Condiciones prohibidas: 1 0 1 0 / 1 0 1 1 / 1 1 0 0 / 1 1 0 1 / 1 1 1 0 / 1 1 1 1	$\emptyset$	$\emptyset$

Fig. 57. *Cuadro de certeza de un circuito duplicador decimal*

Estas operaciones serán realizadas según la naturaleza del último peso decimal del multiplicador. Luego será necesario disponer en la unidad aritmética de circuitos duplicadores, quintuplicadores y que multipliquen por 8 el multiplicando. Examinemos cómo se realizan los dos primeros tipos de circuitos. La figura 57 representa el cuadro de certeza de un circuito duplicador decimal. El código utilizado es el código BCD. La figura 58 representa los diagramas de Karnaugh de X_2 y de C_i. La lectura de los diagramas da:

$$X_2 = \overline{x}_1\, x_2 + x_1\, \overline{x}_4\, \overline{x}_8 + x_2\, \overline{x}_1\, x_4 ,$$

$$C_i = x_8 + x_1\, x_4 + x_4\, x_2 ,$$

$$\overline{C}_i = \overline{x}_4\, \overline{x}_8 + \overline{x}_2\, \overline{x}_1\, \overline{x}_8 ,$$

de donde:

$$X_2 = x_1\, \overline{C}_i + \overline{x}_1\, C_i = x_1 \oplus C_i .$$

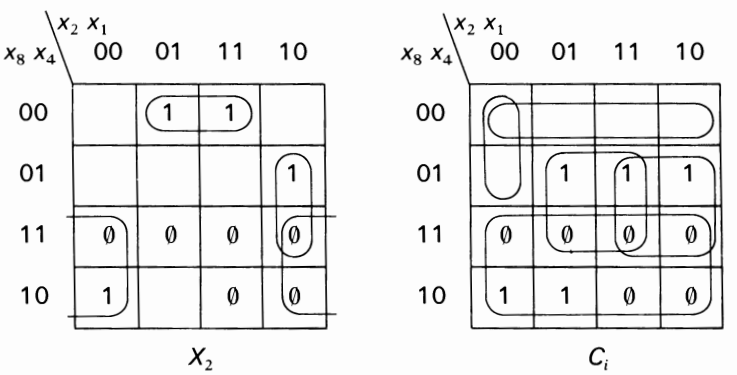


Fig. 58. *Diagramas de Karnaugh de X_2 y C_1*

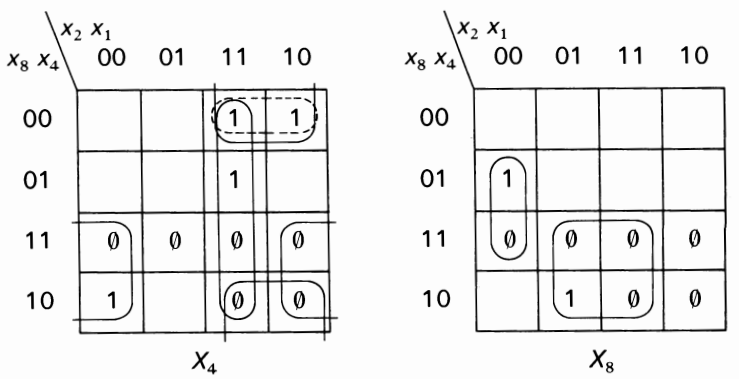


Fig. 59. *Diagramas de Karnaugh de X_4 y X_8*

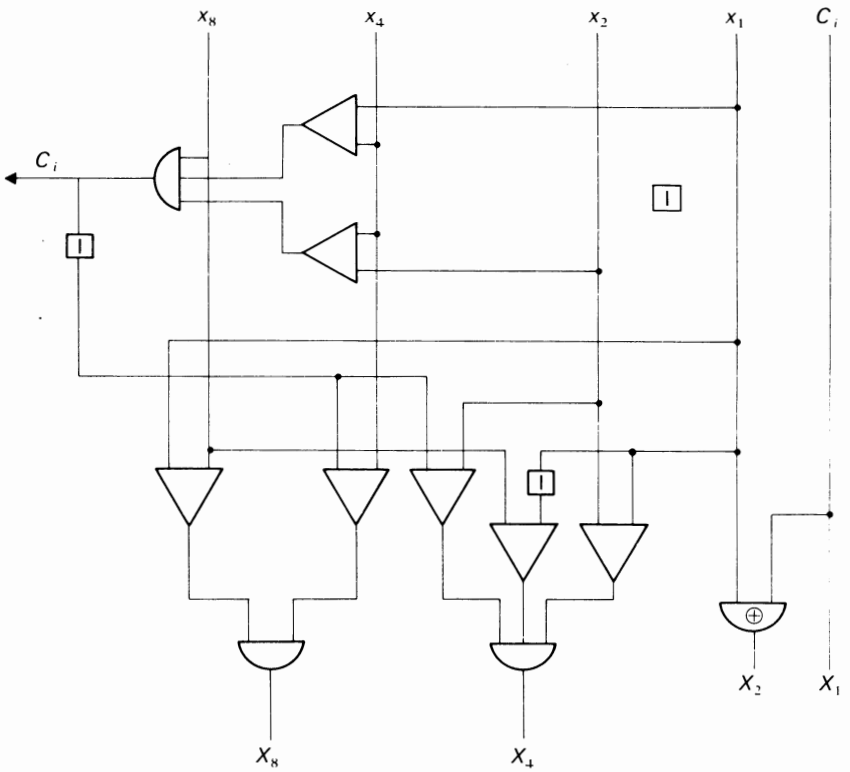


Fig. 60. *Esquema de una etapa duplicadora decimal*

La figura 59 representa los diagramas de Karnaugh de las funciones X_4 y X_8. La lectura de estos diagramas da:

$$X_4 = x_1\, x_2 + x_2\, \overline{x}_4 + \overline{x}_1\, x_8\,,$$

$$X_8 = x_1\, x_8 + \overline{x}_2\, \overline{x}_1\, x_4\,.$$

Obsérvese que X_4 puede escribirse igualmente, considerando el bucle punteado:

$$X_4 = x_1\, x_2 + x_2\, \overline{x}_4\, \overline{x}_8 + \overline{x}_1\, x_8\,.$$

Introducimos voluntariamente un término suplementario X_8 que es redundante. De este modo podremos escribir X_4 bajo la forma:

$$X_4 = x_2\, \overline{C}_i + x_2\, x_1 + x_8\, \overline{x}_1$$

y

$$X_8 = x_4 \overline{C}_i + x_8 x_1 .$$

El circuito duplicador está representado en la figura 60.

Examinemos ahora el caso de un circuito quintuplicador. El cuadro de certeza es el representado en la figura 61.

Obsérvese en esta tabla que:

$$C_4 = x_8$$
$$C_2 = x_4$$
$$C_1 = x_2 .$$

Los arrastres C_4, C_2 y C_1 son engendrados directamente, por los dígitos x_8, x_4 y x_2 de la tetrada. Por otra parte, no hay arrastre C_8 en la tetrada de arrastre:

$$C_8 = 0 .$$

Tetrada de entrada				Tetrada de arrastres				Tetrada de quintuplicador sin arrastre				Tetrada de quintuplicador con arrastre			
x_0	x_4	x_2	x_1	C_8	C_4	C_2	C_1	X_8	X_4	X_2	X_1	Q_8	Q_4	Q_2	Q_1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	0	1	0	1	0	1	0	1
0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1
0	0	1	1	0	0	0	1	0	1	0	1	0	1	1	0
0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	0
0	1	0	1	0	0	1	0	0	1	0	1	0	1	1	1
0	1	1	0	0	0	1	1	0	0	0	0	0	0	1	1
0	1	1	1	0	0	1	1	0	1	0	1	1	0	0	0
1	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0
1	0	0	1	0	1	0	0	0	1	0	1	1	0	0	1
Condiciones prohibidas															
1	0	1	0	∅				∅				∅			
1	0	1	1												
1	1	0	0												
1	1	0	1												
1	1	1	0												
1	1	1	1												

Fig. 61. *Cuadro de certeza de un circuito quintuplicador decimal*

Obsérvese igualmente que la tetrada quíntuple sin arrastre es representada por una sucesión de valores decimales nulos e iguales a 5. Las tetradas nulas tienen lugar cuando $x_1 = 0$ y las iguales a 5, cuando $x_1 = 1$. La salida del circuito quintuplicador será pues una función exclusivamente de x_1, C_1, C_2, y C_4. Los diagramas de Karnaugh correspondientes al cuadro de certeza están representados en la figura 62. La lectura de estos diagramas da:

$$X_1 = \overline{x}_2\, x_1 + x_2\, \overline{x}_1 = x_2 \oplus x_1 = x_1 \oplus C_1 ,$$

$$X_2 = \overline{x}_2\, x_4 + x_2\, x_1\, \overline{x}_4 + x_2\, \overline{x}_1\, x_4 ,$$

$$X_2 = \overline{C}_1\, C_2 + C_1\, \overline{C}_2\, x_1 + C_1\, \overline{C}_2\, \overline{x}_1 ,$$

$$X_4 = \overline{x}_1\, x_8 + x_1\, \overline{x}_8\, x_4 + \overline{x}_2\, x_1\, \overline{x}_8 ,$$

$$X_4 = \overline{x}_1\, C_4 + x_1\, \overline{C}_4\, \overline{C}_2 + x_1\, \overline{C}_1\, \overline{C}_4 ,$$

$$X_8 = x_1\, x_8 + x_1\, x_2\, x_4 = x_1\, C_4 + x_1\, C_1\, C_2 ,$$

$$X_8 = \overline{x}_8\, \overline{x}_4 + \overline{x}_1 + \overline{x}_2\, \overline{x}_8 = \overline{x}_1 + \overline{C}_2\, \overline{C}_4 + \overline{C}_1\, \overline{C}_4 .$$

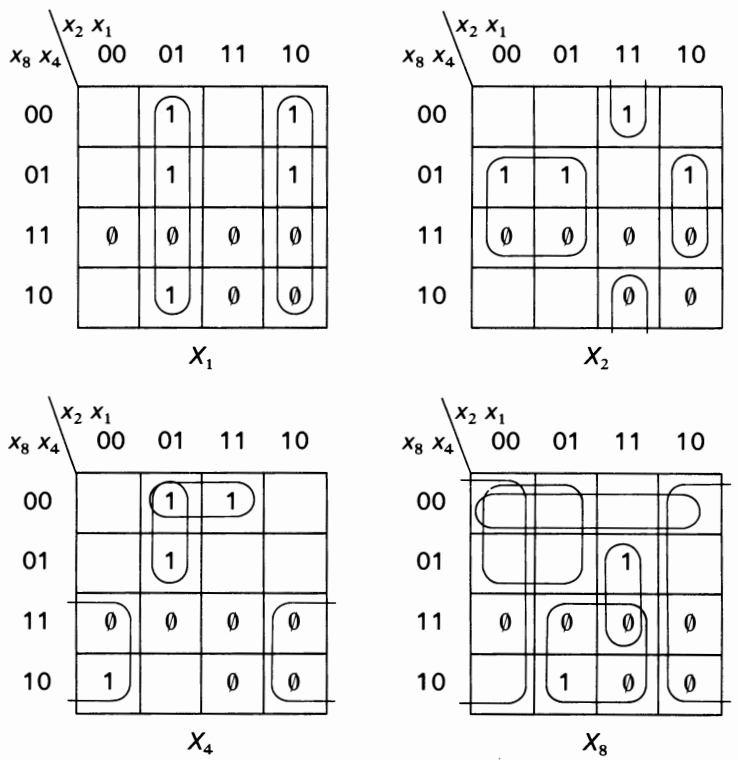


Fig. 62. *Diagramas de Karnaugh de un circuito quintuplicador*

Estas ecuaciones pueden igualmente combinarse entre ellas dando:

$$X_1 = x_1 \oplus C_1$$
$$X_2 = (x_1 C_1 + C_2)\,\overline{X}_8$$
$$X_4 = (x_1 + C_4)\,\overline{X}_8$$
$$X_8 = x_1 C_4 + x_1 C_1 C_2 ,$$

de donde se obtiene el esquema del quintuplicador de la figura 63.

Nótese que este circuito es relativamente simple, debido al hecho de que los arrastres hacia la etapa siguiente son engendrados únicamente por x_8, x_4, y x_2.

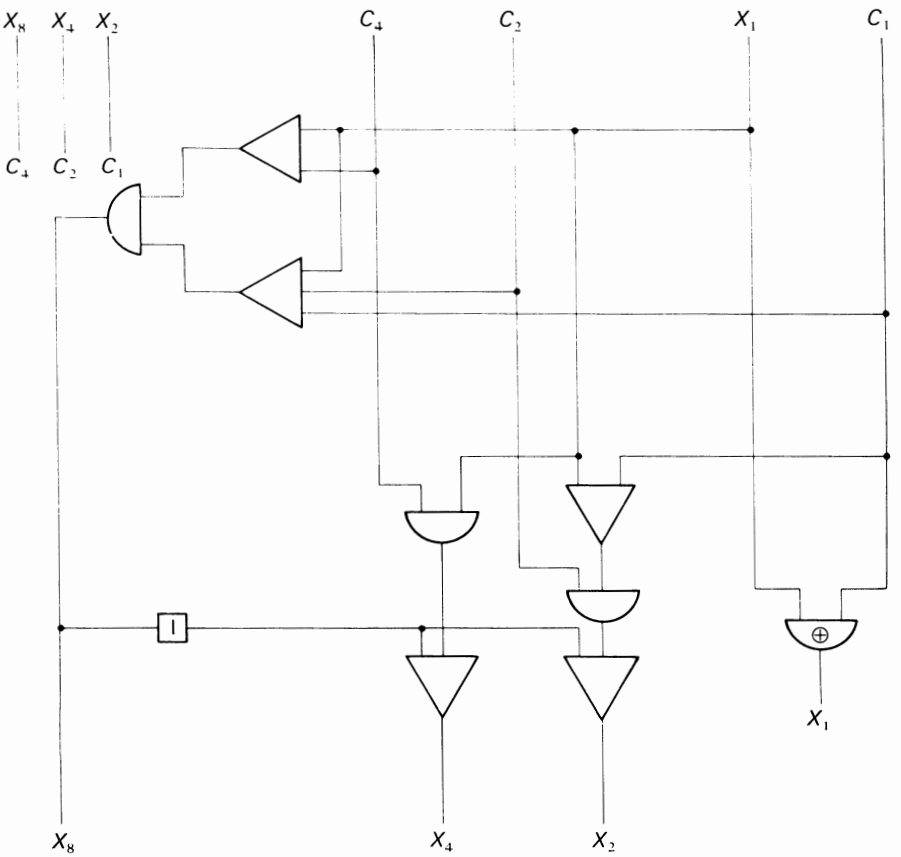


Fig. 63. *Esquema de un circuito quintuplicador*

En el caso de funcionamiento serie-paralelo, estos arrastres son reenviados sobre las entradas correspondientes de la etapa, por medio de tres básculas que memorizarán los arrastres durante el tiempo que separa las dos tetradas. Del mismo modo el lector puede obtener un circuito multiplicador por 8. Le dejamos la libertad de efectuar este ejercicio. En estas condiciones, un multiplicador decimal en su expresión más general, puede ser representado por el esquema de la figura 64. Nótese que a juzgar por la representación de cada dígito decimal por una tetrada, las máquinas en paralelo son muy complejas. Su complejidad aumenta con la rapidez exigida en la ejecución de la multiplicación.

En el esquema de la figura 63, $x_8\ x_4\ x_2$ y x_1 representan el multiplicando, e $y_8\ y_4\ y_2\ y_1$, el multiplicador.

La elaboración de cada producto parcial se efectúa bajo el control de cada dígito decimal sucesivo del multiplicador.

12.4 División decimal

El principio general de la división decimal es similar al de la división binaria. Se sustrae sucesivamente el divisor de los rangos decimales más elevados del dividendo, hasta la obtención de un resto negativo.

A partir de la primera sustracción se inscribe en el registro cociente el valor 0001 que significa que el divisor está comprendido por lo menos una vez en el dividendo.

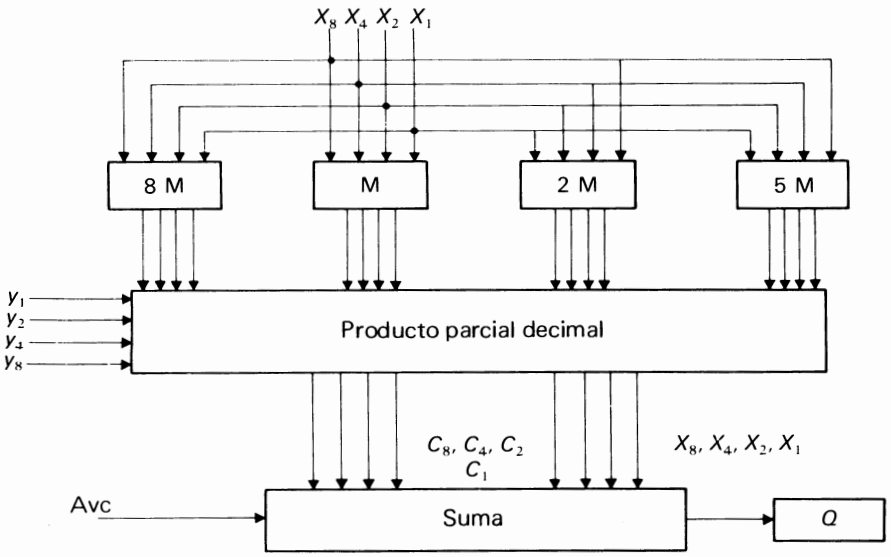


Fig. 64. *Esquema general de un multiplicador decimal*

Seguidamente se sustrae de nuevo el divisor aumentando el cociente en una unidad a cada sustracción. El cociente pasará del valor 0001 al 0010, 0011, etc., que significa que el divisor es contenido 2, 3, 4 veces en el dividendo. De este modo se continúa hasta que la última sustracción dé un resto negativo. En este caso el cociente no se modifica. Se restablece entonces el último resto adicionando el divisor y se desplazan el dividendo y el cociente de una posición decimal (cuatro rangos binarios) a la izquierda. Se efectúa una corrección que consiste en sustraer $6_{10} = 0110$ de la segunda tetrada del dividendo, si existe un arrastre en el orden inmediatamente superior sin modificar el cociente, luego se sustrae de nuevo el divisor tantas veces como sea necesario hasta la obtención de un resto negativo, tomando sucesivamente el cociente los valores 0001, 0010, 0011, etc. Se continúa el proceso hasta el final de la división. Este es señalado por un contador de desplazamientos que disminuye en una unidad a cada desplazamiento y cuyo contenido es nulo al final de la división. Obsérvese que este proceso es muy parecido, salvo en la corrección, al utilizado por las máquinas binarias.

Capítulo 5

Organización de un ordenador

1. Introducción

Un ordenador está destinado a ejecutar una serie de operaciones lógicas y aritméticas según un orden bien determinado. Debe conocer en *cada momento* las operaciones a ejecutar así como los *datos* sobre los cuales se realizan. Se llama *programa* a la sucesión de *instrucciones* necesarias para la ejecución de las operaciones. Estas instrucciones indican en cada momento las operaciones que debe ejecutar el calculador. Este programa es registrado en el calculador antes de su utilización y puede ser modificado según las necesidades y las aplicaciones a tratar. Se concibe pues, que un calculador de *programa registrado* admite una utilización muy flexible. Puede ser utilizado para aplicaciones muy diversas. Para ello bastará con cambiar su programa. Las instrucciones y los datos del programa son almacenados en la memoria del calculador en direcciones sucesivas en forma de palabras de longitud bien determinada. Recibe el nombre de palabra el más pequeño conjunto de bits que puede ser referenciado en la memoria.

2. Formato de una instrucción

Hemos dicho anteriormente que el desarrollo del programa registrado necesita por parte del ordenador la interpretación de un cierto número de instrucciones en secuencia. Luego será necesario examinar ante todo la estructura y la composición de una instrucción. Una instrucción es representada por una palabra de memoria, es decir por una sucesión de 0 y de 1. Los datos necesarios para los cálculos son igualmente almacenados en la memoria en forma de una serie de 0 y de 1 de la misma longitud. El calculador (y el programador) debe poder distinguir la naturaleza diferente de estos dos conceptos. La composición de una instrucción varía según el número de direcciones de la máquina. Así podemos distinguir máquinas de simple dirección, a dos, a tres y a cuatro direcciones. Las dos primeras son utilizadas frecuentemente, sobre todo las del primer tipo. Las máquinas de tres y cuatro direcciones se utilizan muy raramente. Examinemos en primer lugar el formato de una instrucción de una dirección. Esta se presentará bajo la forma de una palabra de memoria dividida en dos partes (figura 1). La parte izquierda está reservada a la identificación de la operación a ejecutar, la parte derecha contiene la dirección del dato sobre el que se realizará la operación.

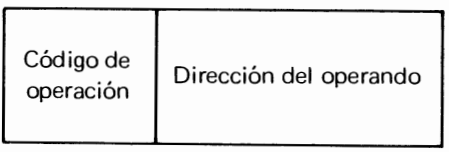


Fig. 1. *Formato de una instrucción de una dirección*

En este caso se supone que uno de los operandos está ya en el acumulador.

La parte izquierda contiene en forma binaria la naturaleza de la operación a realizar. Así, si disponemos de 5 posiciones para el código de operación podremos ejecutar $2^5 = 32$ operaciones distintas. El número de posiciones reservadas para la dirección del operando A dependerá del número de datos almacenados en la memoria. Estos serán representados por una palabra de memoria cuyo formato está representado en la figura 2. La posición del extremo izquierdo de la palabra está reservada al signo, las demás posiciones al módulo del número. Obsérvese que una instrucción o un dato tiene la misma longitud, pero son almacenados en sitios diferentes de la memoria. En efecto, las instrucciones son ejecutadas en secuencia; por lo tanto estarán dispuestas en direcciones crecientes, comenzando por la dirección cero. Los datos ocupan el resto disponible de la memoria. (Fig. 3).

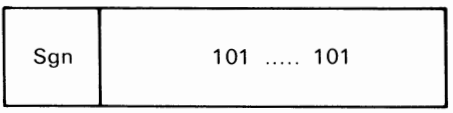


Fig. 2. *Formato de un dato en la memoria*

De este modo se podrán tener programas que contengan muchas instrucciones y pocos datos: éste será el caso de programas de carácter científico. Inversamente se podrán tener pocas instrucciones y muchos datos: éste será el caso de los programas de gestión (Fig. 3).

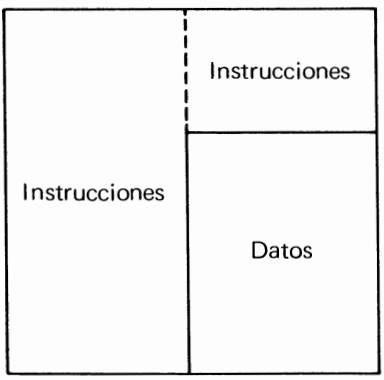


Fig. 3. *Ejemplo de distribución de datos e instrucciones en la memoria*

Volvamos a la composición de la instrucción. Con una palabra de 15 dígitos de los que 5 estarán reservados para el código de operación, tendremos 32 operaciones distintas realizables sobre 1024 direcciones diferentes en la memoria.

2.1 Diferentes categorías de instrucciones

Existen diferentes tipos de instrucciones según la naturaleza de las operaciones que el ordenador deba realizar. Generalmente se distinguen las instrucciones relativas a las operaciones aritméticas, a las operaciones de transferencias, de carga, de desplazamientos y de bifurcación de programas. Finalmente, existen instrucciones para el manejo de índices así como las instrucciones de entradas y salidas destinadas a comunicar el ordenador con sus elementos periféricos. Además del código binario (código de operación) se afecta generalmente a cada instrucción un código nemotécnico, para facilitar al utilizador la interpretación de la instrucción. Examinemos primeramente las instrucciones aritméticas, de transferencias y de carga. La figura 4 da un ejemplo de este tipo de instrucciones. Obsérvese que según este cuadro, las operaciones de transferencias de datos se efectúan conservando el contenido del registro (o de la memoria) que contiene el dato antes de la transferencia. También para las operaciones aritméticas uno de los operandos está siempre en el acumulador y en el registro Q antes de la operación. El resultado aparece siempre en el acumulador. Este registra todos los resultados parciales en el curso del programa.

Para las operaciones de desplazamiento, el número de posiciones a desplazar está indicado en la parte dirección de la instrucción.

Instrucción	Código nemotécnico	Descripción de la operación
Adición	A	Adicionar al contenido del acumulador el número contenido en memoria en la dirección dado por la instrucción. El resultado aparece en el acumulador.
Puesta a cero y adición	CLA	Puesta a cero del acumulador y adición al contenido del acumulador del número registrado en la memoria en la dirección dada por la instrucción. Esta operación equivale a introducir el contenido de la memoria en el acumulador.
Sustracción	S	Sustraer del contenido del acumulador el número memorizado en la dirección dada por la instrucción. El resultado de la operación aparece en el acumulador.
Puesta a cero y sustracción	CLS	Puesta a cero del acumulador y sustracción del contenido del acumulador del número memorizado en la dirección dada por la instrucción. El resultado aparece en el acumulador. Esta operación equivale a introducir el contenido de la memoria cambiado de signo en el acumulador.

Instrucción	Código nemotécnico	Descripción de la operación
Complementación del módulo	COM	Complementar el módulo del número en el acumulador. El signo no varía.
Almacenamiento del acumulador	STA	El contenido del acumulador es almacenado en la memoria en la dirección indicada por la instrucción. El contenido del acumulador no varía.
Almacenamiento del registro Q	STQ	El contenido del registro Q es almacenado en la memoria en la dirección indicada por la instrucción. El contenido del registro Q no varía.
Almacenamiento de ceros	STZ	El contenido memorizado en la dirección dada por la instrucción es reemplazado por ceros (signo positivo).
Carga del registro Q	LDQ	El contenido memorizado en la dirección dada por la instrucción es cargado en el registro Q. Este contenido no varía.
Multiplicación	M	Multiplicar el contenido memorizado en la dirección dada por la instrucción, por el contenido del registro Q. El resultado aparece en el acumulador-registro Q. El contenido de Q se pierde.
División	D	Dividir el contenido del conjunto acumulador-registro Q por el número memorizado en la dirección dada por la instrucción. El cociente aparece en el registro Q, el resto en el acumulador.
Desplazamiento a la izquierda del acumulador	SLA	Desplazar el contenido del acumulador conservando el signo un cierto número de posiciones indicado en la parte dirección de la instrucción. Los bits de la izquierda del acumulador se pierden.
Desplazamiento a la derecha del acumulador	SRA	Desplazar el contenido del acumulador conservando el signo un cierto número de posiciones a la derecha indicando en la parte di rección de la instrucción. Los bits de la derecha del acumulador se pierden.
Desplazamiento total a la izquierda del conjunto acumulador-registro Q	SLL	Desplazar a la izquierda el contenido del registro Q al acumulador. El contenido del registro Q reemplaza al contenido del acumulador. El signo del registro Q reemplaza al signo del acumulador.

Instrucción	Código nemotécnico	Descripción de la operación
Desplazamiento total a la derecha del conjunto acumulador-registro Q	SRL	Desplazar a la derecha el contenido del acumulador al registro Q. El contenido del acumulador reemplaza al contenido del registro Q. El signo del acumulador sustituye al signo del registro Q.
Desplazamiento a la izquierda del registro Q	SLQ	Desplazar a la izquierda el contenido del registro Q un cierto número de posiciones indicadas en la parte dirección de la instrucción. El signo es conservado. Los bits de la izquierda del registro Q se pierden.
Desplazamiento a la derecha del registro Q	SRQ	Desplazar a la derecha el contenido del registro Q un cierto número de posiciones indicadas en la parte dirección de la instrucción. El signo se conserva. Los bits de la derecha del registro Q se pierden.
"Y"	AND	Multiplicación lógica de cada bit del acumulador con cada bit de una palabra memorizada en la dirección dada por la instrucción. El resultado aparece en el acumulador.
"O"	OR	Adición lógica de cada bit del acumulador con cada bit de una palabra memorizada en la dirección dada por la instrucción. El resultado aparece en el acumulador.
"O exclusivo"	ORX	"O exclusivo" (suma sin arrastre) entre el contenido del acumulador y una palabra memorizada en la dirección dada por la instrucción. El resultado aparece en el acumulador.

Fig. 4. *Instrucciones aritméticas y de desplazamientos*

El número de posiciones a desplazar es introducido antes de la operación en un contador de desplazamientos que disminuye en una unidad a cada desplazamiento. Cuando su contenido es nulo, termina la operación de desplazamiento.

En la mayor parte de los programas, es necesario efectuar operaciones iterativas, lo que se traduce por la obligación de efectuar una bifurcación del programa para repetir sucesivamente un cierto número de operaciones idénticas (bucle de programa). También es frecuentemente necesario romper la secuencia del programa en función de ciertos resultados intermedios. Entonces se efectúa una bifurcación condicional del programa. Finalmente en algunos casos es necesario efectuar una bifurcación incondicional del programa. Entonces se pasa de un punto a otro del programa sin condición previa. El cuadro de la figura 5 da un ejemplo de las bifurcaciones de programa que se encuentran en la práctica.

Estas instrucciones de bifurcación entran en una categoría más general de instrucciones llamadas instrucciones de *"control"*. Entre estas instrucciones se encuentran "parada y arranque", "parada y bifurcación" y "no operación".

Instrucción	Código nemotécnico	Descripción de la operación
Bifurcación incondicional	JMP	Bifurcación a la instrucción cuya dirección está indicada en la parte dirección de la instrucción JMP.
Bifurcación si el contenido del acumulador es cero	JAZ	Si (A) = 0 la instrucción siguiente es la instrucción memorizada en la dirección indicada en la parte dirección de la instrucción. En caso contrario se continúa en secuencia.
Bifurcación si el contenido del acumulador es diferente de cero	JAN	Si (A) $\neq$ 0, la instrucción siguiente es la instrucción memorizada en la dirección indicada en la parte dirección de la instrucción. En caso contrario se continúa en secuencia.
Bifurcación si el contenido del acumulador es positivo	JAP	Si (A) $>$ 0, la instrucción siguiente es la instrucción memorizada en la dirección indicada en la parte dirección de la instrucción. En caso contrario se continúa en secuencia.
Bifurcación si el contenido del acumulador es negativo	JAM	Si (A) $<$ 0, la instrucción siguiente es la instrucción memorizada en la dirección indicada en la parte dirección de la instrucción. En caso contrario se continúa en secuencia.
Bifurcación si el contenido del acumulador es par	JAE	Si (A) es par, la instrucción siguiente es la instrucción memorizada en la dirección indicada en la parte dirección de la instrucción. En caso contrario se continúa en secuencia.
Bifurcación si el contenido del acumulador es impar	JAQ	Si (A) es impar, la instrucción siguiente es la instrucción memorizada en la dirección indicada en la parte dirección de la instrucción.
Bifurcación si el contenido del registro Q es positivo	JQP	Si (Q)$>$0, la instrucción siguiente es la instrucción memorizada en la dirección indicada en la parte dirección de la instrucción. En caso contrario se continúa en secuencia.

Instrucción	Código nemotécnico	Descripción de la operación
Bifurcación si el contenido del registro Q es negativo	JQM	Si (Q)<0, la instrucción siguiente es la instrucción memorizada en la dirección indicada en la parte dirección de la instrucción. En caso contrario se continúa en secuencia.
Bifurcación si el contenido del registro Q es cero	JQZ	Si (Q) = 0, la instrucción siguiente es la instrucción memorizada en la dirección indicada en la parte dirección de la instrucción. En caso contrario se continúa la secuencia.
Bifurcación en caso de Over-flow (rebasamiento de capacidad)	JOV	Si B Overflow = 1, la instrucción siguiente es la instrucción memorizada en la dirección indicada en la parte dirección de la instrucción. En caso contrario continuar en secuencia.
Parada del ordenador	HLT	Parada del calculador.
Parada y arranque	HPR	Parada del calculador seguido de un nuevo arranque en la instrucción siguiente habiendo apoyado el interruptor "arranque" sobre el pupitre.
Parada y bifurcación	HJP	Parada del calculador y bifurcación posterior a la instrucción memorizada en la dirección indicada en la parte dirección de la instrucción habiendo apoyado el interruptor "arranque".
No operación	NOP	Ejecución de la instrucción siguiente.

Fig. 5. *Instrucciones de control*

3. Modificación de instrucciones

Una gran ventaja de las máquinas de calcular consiste en la posibilidad de modificar una instrucción en el curso del programa sin modificar la instrucción almacenada en memoria, esta última permanece invariable en tanto que el programa no sea modificado. La modificación de una instrucción concierne principalmente a su parte dirección. Esta modificación es efectuada por una indexación *directa* o *indirecta*. El principio de la indexación de una dirección es muy simple: se añade o se quita de la dirección de la instrucción el contenido de un registro especial llamado registro índice. *La nueva dirección calculada de este modo es la dirección efectiva de la instrucción en ejecución.*

Obsérvese que la dirección almacenada en memoria no varía.

La gran ventaja de la indexación reside en el hecho de que el contenido del registro índice puede ser incrementado o disminuido entre dos instrucciones en cierta cantidad. De este modo una misma instrucción puede ser ejecutada sobre dos números diferentes memorizados en direcciones sucesivas obtenidas por la operación de indexación.

Instrucción	Código nemotécnico	Descripción de la operación
Indexación	SXR	El índice, es decir el contenido de la parte dirección de la instrucción es introducido en el registro índice.
Carga del registro índice	LDX	El contenido memorizado en la dirección indicada en la parte dirección de la instrucción es introducido en el registro índice.
Almacenamiento del registro índice	STX	El contenido del registro índice es memorizado en la dirección indicada en la parte dirección de la instrucción. El contenido del registro índice no varía.
Bifurcación según índice	JPX	Si el contenido del registro índice es mayor que el decremento D, se disminuye el contenido de este registro en D y se bifurca a la instrucción indicada en la parte dirección de la instrucción. En caso contrario se continúa en secuencia
Bifurcación con incremento del índice	JIX	El contenido del registro índice es incrementado en D y se bifurca a la instrucción indicada en la parte dirección de la instrucción.

Fig. 6. *Instrucciones de indexación*

Generalmente existen cuatro instrucciones que hacen posible la indexación: la instrucción de indexación que introduce en el registro índice el contenido de su parte dirección, las instrucciones de almacenamiento y de carga, las instrucciones de bifurcación según el contenido del índice. La instrucción de almacenamiento tiene por efecto memorizar el contenido del registro índice. La instrucción de carga introduce en el registro índice el contenido de una dirección de memoria. Se distinguen dos condiciones de bifurcación según que el índice sea incrementado o disminuido. En el caso de que el índice sea disminuido en una cantidad D, la instrucción de bifurcación disminuye en D el contenido del registro índice si su contenido es mayor que D, bifurcando después a la dirección contenida en la parte dirección de la instrucción. Si el contenido del registro índice es igual o inferior a D, se continúa en secuencia. En el caso de que el índice se incremente en D, la

instrucción de bifurcación aumenta el contenido del registro índice en D y bifurca incondicionalmente a la dirección contenida en su parte dirección. En la mayor parte de las máquinas $D = 1$. En algunas máquinas, existen varios registros índices. La instrucción debe pues poder seleccionar uno de estos registros e indicar el valor del decremento en el caso en que varios decrementos sean utilizados (instrucción de bifurcación). Además de las instrucciones de indexación, cualquier otra instrucción puede ser indexada. El formato de una instrucción indexada está representado en la figura 7.

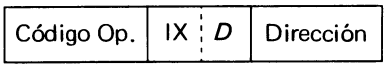


Fig. 7. *Formato de una instrucción indexada*

Así, la instrucción

Add 0 1 0 0 1 0 0 1

es una instrucción indexada. La máquina comprende tres registros índices 01, 10 y 11. La dirección efectiva de la instrucción es la dirección 00 1001 a la que se ha añadido (o quitado) el contenido del registro índice 01.

3.1 Direccionamiento indirecto

Otro modo de modificar la parte dirección de una instrucción consiste en utilizar el *direccionamiento indirecto.* En este caso la dirección de la instrucción indexada no es la dirección real, sino la dirección de la dirección. De este modo se puede modificar por medio del cálculo el contenido de una dirección de memoria que se convierte en la dirección efectiva de una instrucción en curso.

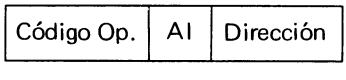


Fig. 8. *Formato de una instrucción con direccionamiento indirecto*

En la figura 8 se representa el formato de una instrucción que utiliza el direccionamiento indirecto. La posición AI contendrá un 1 en el caso de un direccionamiento indirecto y un 0 en caso contrario. La máquina interpretará automáticamente la instrucción y obtendrá la dirección efectiva de la instrucción a ejecutar.

4. Instrucciones de entradas y salidas

Estas instrucciones están destinadas a comunicar el calculador con sus órganos periféricos. Son propias de cada tipo de instalación. No solamente deben controlar la lectura y escritura de los datos sino también dar las órdenes necesarias a los

elementos periféricos (funcionamiento de las bobinas, de las cintas perforadas o de los martillos de impresión en el caso de una impresora). Los movimientos de los datos y los mandos de los elementos periféricos están generalmente bajo el control de un supervisor (operating system). Por el momento nos limitaremos a las instrucciones de arranque y final de las operaciones de entradas y de salidas.

Estas instrucciones corresponden a la lectura de los datos a partir de un elemento periférico determinado. La determinación del periférico está indicada en la parte dirección de cada instrucción. Existen tres tipos de instrucciones: dos para la selección (lectura y escritura) del periférico correspondiente y otra la llamada instrucción de "copia" (Fig. 9).

Instrucción	Código nemotécnico	Descripción de la operación
Selección para la lectura	RDS	Selección del elemento periférico para la lectura según el contenido de la parte dirección de la instrucción.
Selección para la escritura	WRS	Selección del elemento periférico para la escritura según el contenido de la parte dirección de la instrucción.
Orden de lectura o de escritura	CPY	Lectura o escritura con el elemento periférico correspondiente.

Fig. 9. *Instrucciones de entradas-salidas*

5. Máquinas de varias direcciones

Acabamos de examinar el caso de máquinas con una sola dirección. En este tipo de máquinas la instrucción solamente contiene una dirección. La figura 8 da un ejemplo de una instrucción de direccionamiento indirecto de una sola dirección. Existen otros tipos de máquinas que utilizan dos, tres e incluso cuatro direcciones. Las máquinas más comunes son las de una dirección. La figura 10 representa el formato de una instrucción a doble, a triple y a cuádruple dirección.

El inconveniente principal de las máquinas de direcciones múltiples está en la mayor longitud de las palabras a tratar y como consecuencia una mayor complejidad de los circuitos y *un volumen más grande de las memorias.* A continuación nos limitaremos al caso de las máquinas de una dirección.

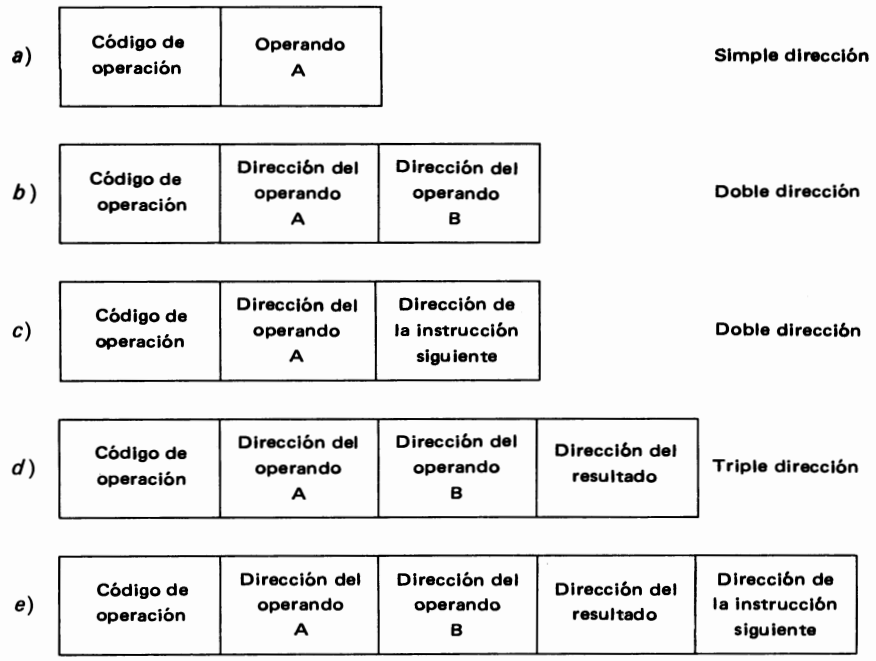


Fig. 10. *Diferentes formatos de las palabras de una instrucción*

6. Concepción de un calculador

La concepción de un calculador depende de su utilización específica. La concepción general de un calculador industrial será diferente de un calculador científico o de gestión. La diferencia será mucho mayor en lo que concierne a los elementos periféricos. La estructura de un calculador es afectada en gran manera por la estructura de la instrucción. En efecto, según que se trate de una máquina de simple dirección o de direcciones múltiples o de una máquina a palabras o a caracteres, la naturaleza de los circuitos del hardware serán muy diferentes. A continuación, nos limitaremos al caso de las máquinas binarias a palabras. Primeramente examinaremos el caso más simple, el de una máquina de una sola dirección.

6.1 Calculador de simple dirección

La instrucción de un calculador de simple dirección está representada en la figura 10. El esquema general de un calculador de este tipo en la figura 11.

Supongamos que las instrucciones pueden ser indexadas por medio de tres registros de índices I_1, I_2 e I_3. Una matriz de selección elige el registro según el

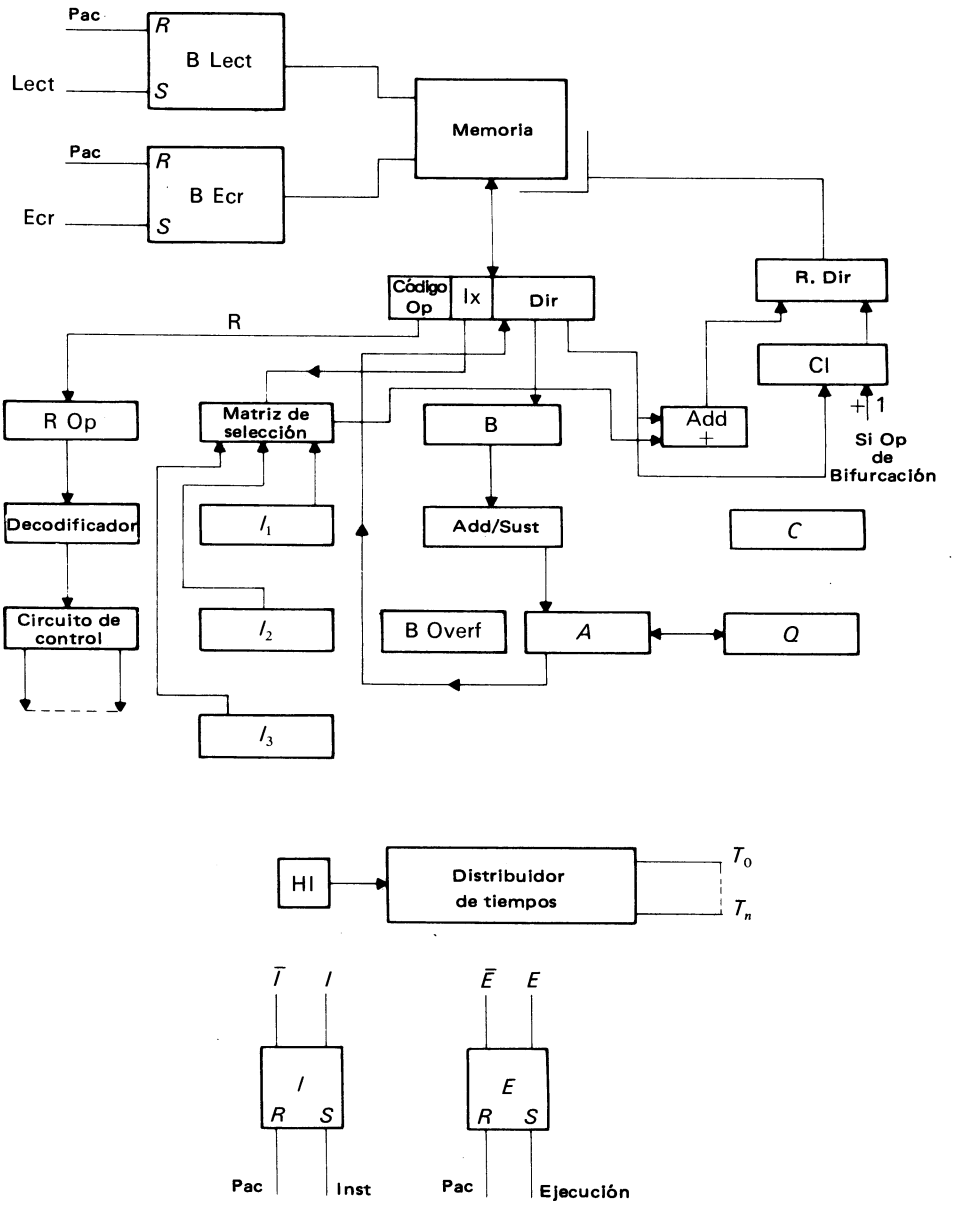


Fig. 11. *Esquema general de un calculador de simple dirección con posibilidad de indexación de las instrucciones*

contenido de la parte índice de la instrucción. La memoria es controlada por dos básculas "lectura" y "escritura". La selección de una palabra de memoria se efectúa por medio del registro de dirección "R Dir" que recibe la dirección de la instrucción a partir del contador de instrucciones *CI*. En el caso de una instrucción indexada la dirección efectiva se obtiene por adición del contenido de uno de los registros de índices I_1 I_2 ó I_3 a la dirección de la instrucción. Esta dirección efectiva es introducida a continuación en el registro de dirección por la lectura del dato. La memoria está conectada en los dos sentidos al registro de memoria. Durante la lectura de una instrucción este registro recibe la instrucción a ejecutar. La parte "código de operación" es introducida en un registro operación R Op, el cual está conectado a un decodificador que interpreta la operación a ejecutar. La salida del decodificador está conectada a los circuitos de control de las operaciones que suministran las señales necesarias para la ejecución de la operación. El contador de instrucción CI aumenta su contenido en una unidad a cada lectura de una instrucción con el fin de preparar la lectura de la instrucción siguiente. Una vez interpretada y decodificada la instrucción, el registro de dirección contiene la dirección del dato de la instrucción. Un segundo ciclo de lectura introduce este dato en el registro de memoria, a continuación este dato es transferido al registro *B* para efectuar el cálculo. Se supone que el otro dato está ya en el acumulador. Según la naturaleza del cálculo (multiplicación o división) el registro *Q* será o no utilizado. El resultado aparece en el acumulador y eventualmente en el registro *Q*. Un contador *C* indica el número de desplazamientos necesarios, siendo nulo su contenido al final de la operación. La operación a realizar generalmente se divide en dos ciclos: un ciclo de instrucción y un ciclo de ejecución. Durante el ciclo de instrucción la máquina interpreta la instrucción y prepara los circuitos para la ejecución de la operación. Al mismo tiempo aumenta el contenido del contador de instrucciones en una unidad para preparar la lectura de la instrucción siguiente. Durante el ciclo de ejecución introduce los datos necesarios a la operación en los diferentes registros y ejecuta la operación. La división en dos ciclos es realizada por medio de dos básculas I y E que se ponen en 1 al principio del ciclo de instrucción y de ejecución. Los impulsos necesarios para efectuar las operaciones de cada ciclo son suministrados por un distribuidor de tiempos, el cual está controlado por el reloj del calculador. El número de tiempos elementales es impuesto por la operación más complicada a realizar en uno de los dos ciclos. Esto significa que para ciertas operaciones todos los impulsos del distribuidor de tiempos no serán utilizados. Obsérvese igualmente que la ejecución de una operación aritmética necesita dos ciclos de memoria: uno para la lectura y la reescritura de la instrucción, y el otro para la lectura y la reescritura del dato. Sin embargo, ciertas operaciones como las bifurcaciones y los desplazamientos solamente necesitan un ciclo de memoria.

6.2 Máquinas de dos direcciones

Examinemos ahora el caso particular de una máquina de dos direcciones. La instrucción tendrá el formato definido en 10_c. Contendrá un código de operación, eventualmente una parte reservada a la indexación de la instrucción, la dirección del operando A y la dirección de la instrucción siguiente. En las máquinas de este tipo se economiza el contador de instrucciones ya que cada instrucción contiene la dirección de la instrucción siguiente. En la figura 12 está representado el esquema general de un calculador de este tipo.

Obsérvese no obstante que la utilización de una segunda dirección tiene como consecuencia una mayor ocupación de la memoria así como la utilización de palabras de mayor longitud, lo que provoca un aumento de la complejidad de los circuitos del hardware. Por esto se prefiere generalmente las máquinas de simple dirección.

7. Circuitos de control de las micro-operaciones

Los circuitos de control de las micro-operaciones necesarias a la ejecución de la operación dependen del decodificador de operación, de las básculas I y E según el ciclo de la máquina, del distribuidor de tiempos T_0 a T_n y eventualmente del contador C. A continuación vamos a examinar la naturaleza de las micro-operaciones que el calculador debe efectuar para la ejecución de algunas operaciones simples. El cuadro de las figuras 13 a 16 refleja todos los detalles de las operaciones de carga del acumulador, de almacenamiento de un número en la memoria, de la adición y de la sustracción de los números. La parte derecha de este cuadro representa la lista de las micro-instrucciones necesarias para efectuar la operación. Para simplificar la explicación supongamos que la máquina es de una sola dirección y que las instrucciones no son indexadas. Obsérvese que todas las operaciones del ciclo de instrucción y de ejecución se efectúan en cuatro tiempos elementales T_0 a T_3. El distribuidor de tiempos, controlado por el reloj, suministra los cuatro impulsos T_0 a T_3. Nótese también que para las cuatro operaciones estudiadas el ciclo de instrucción es el mismo. Veremos más adelante que es diferente para el caso de las operaciones de bifurcación y de desplazamientos. No obstante se intentará siempre tener el mayor número de ciclos de instrucción idénticos, es posible realizar máquinas en las que todos los ciclos I son efectivamente iguales. Durante el ciclo de instrucción la máquina efectúa las micro-operaciones siguientes:

– lectura de la instrucción,
– transferencia del código de operación al registro de operación,
– aumento del contenido del contador de instrucciones en una unidad,
– transferencia de la dirección de la instrucción al registro de direcciones,
– paso al ciclo de ejecución,

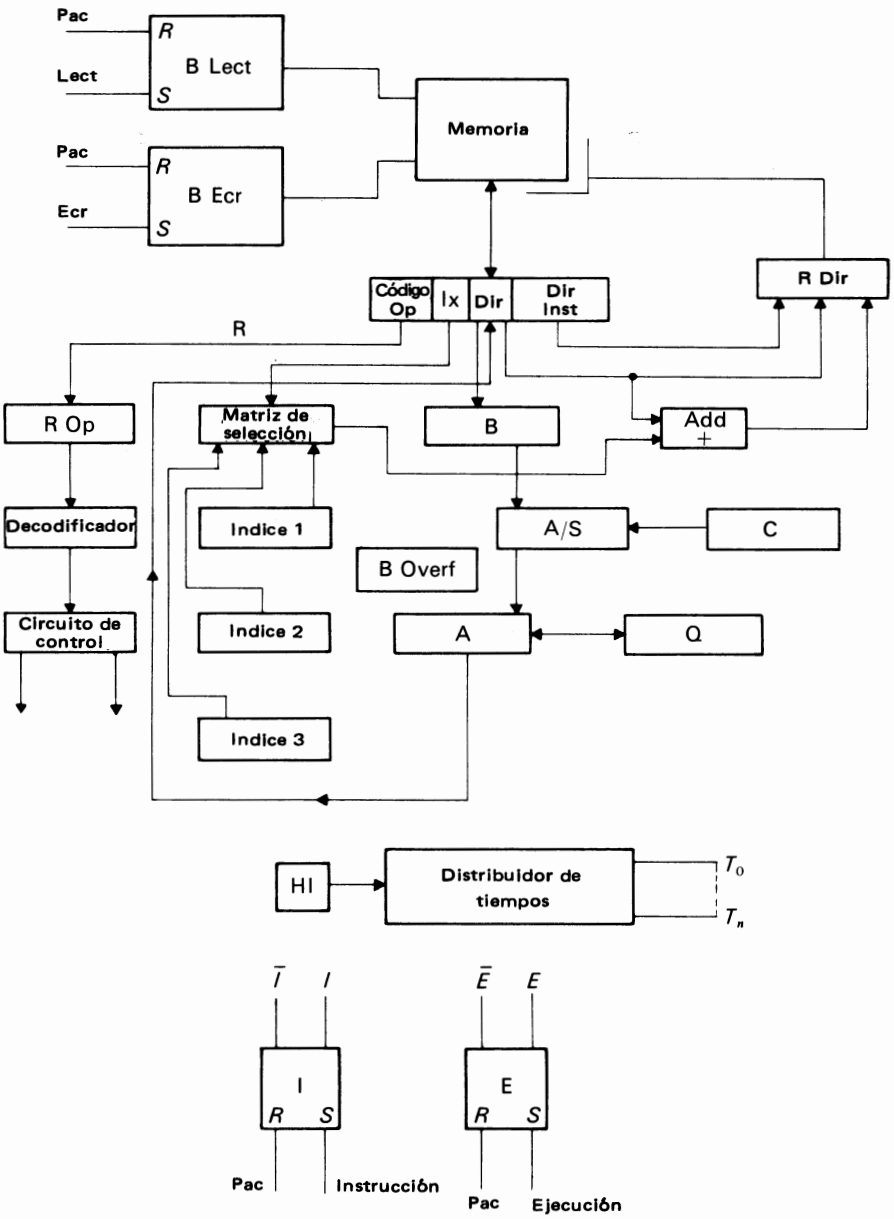


Fig. 12. *Esquema general de un calculador a doble dirección*

Tiempos elementales	Micro-operaciones	Detalle de las micro-operaciones
IT_0	1 ⇒ B Lect (Mem ⇒ R)	Lectura de la instrucción. La puesta en 1 de la báscula Lect provoca la transferencia de la instrucción al registro de memoria R.
IT_1	(Op) R ⇒ R Op 0 ⇒ B Lect	El código de operación de la instrucción es transferido al registro de operación. La báscula Lect es puesta a cero.
IT_2	(CI) + 1 ⇒ CI	El contenido del contador de instrucción aumenta en una unidad para preparar la lectura de la instrucción siguiente.
IT_3	(Dir) R ⇒ R Dir 1 ⇒ E, 0 ⇒ 1	La dirección de la instrucción es transferida al registro de direcciones. Puesta en 1 de la báscula E. Puesta a cero de la báscula I. La máquina entra en el ciclo de ejecución.
ET_0	1 ⇒ B Lect (Nem ⇒ R)	Lectura del contenido de la memoria en la dirección contenida en el registro de direcciones (datos). El dato es transferido al registro de memoria R.
ET_1	(R) ⇒ B 0 ⇒ B Lect	El contenido del registro de memoria es transferido al registro B. La báscula Lect es puesta a cero.
ET_2	(A) + (B) ⇒ A 1 ⇒ B Overflow si $C_1 + C_2 = 1$	El contenido de B es adicionado al contenido del acumulador. El resultado aparece en el acumulador. Una báscula B Overflow es puesta en 1 si hay rebasamiento de capacidad del acumulador.
ET_3	(CI) ⇒ R Dir 0 ⇒ E, 1 ⇒ I	El contenido del contador de instrucción es transferido al registro de direcciones. Este contiene la dirección de la instrucción siguiente. La báscula I es puesta en 1, la báscula E en 0. La máquina entra en el ciclo de instrucción.

Fig. 13. *Operación de adición*

Tiempos elementales	Micro-operaciones	Detalle de las micro-operaciones
IT_0	1⇒ B Lect (Mém ⇒ R)	Lectura de la instrucción. La puesta en 1 báscula Lect provoca la transferencia de la instrucción en el registro de memoria R.
IT_1	(Op) R ⇒ R Op 0 ⇒ B Lect	El código de operación de la instrucción es transferido al registro de operación. La báscula Lect es puesta a cero.
IT_2	(CI) + 1 = CI	El contenido del contador de instrucción aumenta en una unidad para preparar la lectura de la instrucción siguiente.
IT_3	(Dir) R ⇒ R Dir 1 ⇒ E, 0 ⇒ I	La dirección de la instrucción es transferida al registro de dirección. Puesta en 1 de la báscula E. Puesta a cero de la báscula I. La máquina entra en el ciclo de ejecución.
ET_0	1 ⇒ B Lect (Mém ⇒ R)	Lectura del contenido de la memoria en la dirección contenida en el registro de direcciones. El dato es transferido al registro de memoria R.
ET_1	(R) ⇒ B, 0 ⇒ B Lect 0 ⇒ A	El contenido del registro de memoria es transferido al registro B, la báscula Lect es puesta a cero así como el acumulador. Adicionando seguidamente el contenido de B al del acumulador, tendremos B en el acumulador.
ET_2	(B) + (A) ⇒ A	El contenido de B es adicionado al contenido del acumulador. El resultado aparece en el acumulador.
ET_3	(CI) ⇒ R Dir 0 ⇒ E, 1 ⇒ I	El contenido del contador de instrucción es transferido al registro de direcciones. Este contiene entonces la dirección de la instrucción siguiente. La báscula I es puesta a 1, la báscula E en 0. La máquina entra en el ciclo de instrucción.

Fig. 14. *Operación de carga del acumulador*

Tiempos elementales	Micro-operaciones	Detalle de las micro-operaciones
IT_0	1 ⇒ B Lect (Mem ⇒ R)	Lectura de la instrucción. La puesta en 1 de la báscula Lect provoca la transferencia de la instrucción al registro de memoria R.
IT_1	(Op) R ⇒ R Op 0 ⇒ B Lect	El código de operación de la instrucción es transferido al registro de operación. La báscula Lect es puesta a cero.
IT_2	(CI) + 1 ⇒ CI	El contenido del contador de instrucción aumenta en una unidad para preparar la lectura de la instrucción siguiente.
IT_3	(Dir) R ⇒ R Dir 0 ⇒ I, 1 ⇒ E	La dirección de la instrucción es transferida al registro de dirección. Puesta en 1 de la báscula E. Puesta a cero de la báscula I. La máquina entra en el ciclo de ejecución.
ET_0	1 ⇒ B Lect (Mem ⇒ R)	Lectura del contenido de la memoria en la dirección contenida en el registro de direcciones. El dato es transferido al registro de memoria R.
ET_1	(R) ⇒ B, 0 ⇒ B Lect	El contenido del registro de memoria es transferido al registro B. La báscula Lect es puesta a cero.
ET_2	(A) – (B) ⇒ A, 1 ⇒ B Overflow si $C_1 + C_2 = 1$	El contenido del registro B es sustraído del contenido del acumulador. El resultado aparece en el acumulador. Una báscula B Overflow es puesta en 1 si hay rebasamiento de capacidad en el acumulador.
ET_3	(CI) ⇒ R Dir 0 ⇒ E, 1 ⇒ I	El contenido del contador de instrucciones es transferido al registro de direcciones. Este contiene entonces la dirección de la instrucción siguiente. La báscula I es puesta en 1, la báscula E en 0. La máquina entra en el ciclo de instrucción.

Fig. 15. *Operación sustracción*

Tiempos elementales	Micro-operaciones	Detalle de las micro-operaciones
IT_0	1 ⇒ B Lect (Mem ⇒ R)	Lectura de la instrucción. La puesta en 1 de la báscula Lect transfiere la instrucción al registro de memoria R.
IT_1	(Op) R ⇒ R Op 0 ⇒ B Lect	El código de operación de la instrucción es transferido al registro de operación. La báscula Lect es puesta a cero.
IT_2	(CI) + 1 ⇒ CI	El contenido del contador de instrucción aumenta en una unidad para preparar la lectura de la instrucción siguiente.
IT_3	(Dir) R ⇒ R Dir 0 ⇒ I, 1 ⇒ E	La parte dirección de la instrucción es transferida al registro de direcciones. La báscula E es puesta en 1, la báscula I en 0. La máquina entra en el ciclo de ejecución.
ET_0	(A) ⇒ R	Transferencia del contenido del acumulador al registro de memoria.
ET_1	1 ⇒ B Escrit	Escritura en memoria (A) ⇒ R.
ET_2	0 ⇒ B Escrit	Puesta en cero de la báscula B Escrit.
ET_3	(CI) ⇒ R Dir 0 ⇒ E, 1 ⇒ I	El contenido del contador de instrucción es transferido al registro de direcciones. Este contiene entonces la dirección de la instrucción siguiente. La báscula E en 0. La máquina entra en el ciclo de ejecución.

Fig. 16. *Operación de almacenamiento en memoria del contenido del acumulador*

Al principio de la operación la máquina está en el ciclo de instrucción ($1 \Rightarrow I$) y la dirección de la primera instrucción es la 0000. Recuérdese que cada lectura de la memoria es seguida de una operación de escritura con el fin de conservar la instrucción o el dato en la memoria.

El ciclo de ejecución varía según la naturaleza de las operaciones a realizar. La más compleja es la división de dos números. La figura 16 representa el esquema de realización del ciclo de instrucción, la figura 17 sus circuitos lógicos. En la figura 18 se representa el esquema general del calculador.

En el ejemplo que acabamos de tratar, el calculador solamente efectúa cuatro operaciones elementales. Se trata pues de un calculador muy simplificado. Hemos visto que el ciclo de instrucción es el mismo para las cuatro operaciones. Si añadimos operaciones suplementarias tales como las operaciones de bifurcación y desplazamientos, con la estructura del calculador elegida, será necesario modificar el ciclo de instrucción.

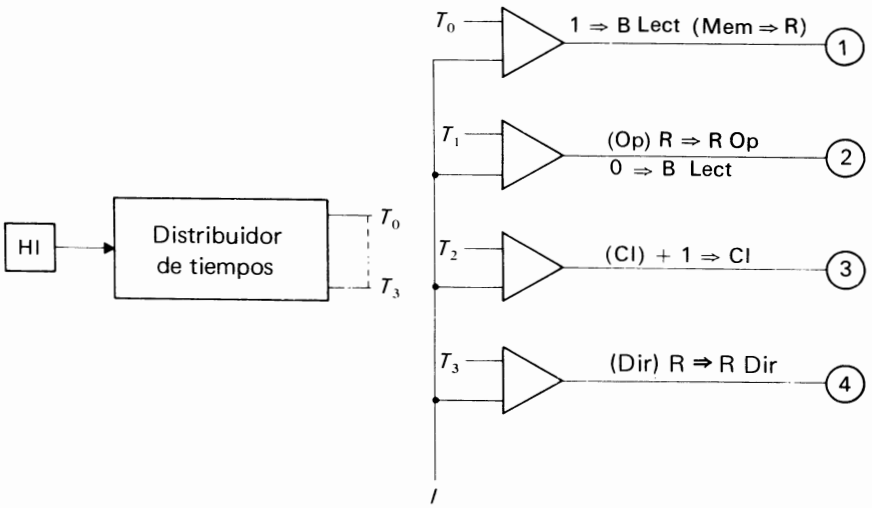


Fig. 17. *Realización del ciclo de instrucción*

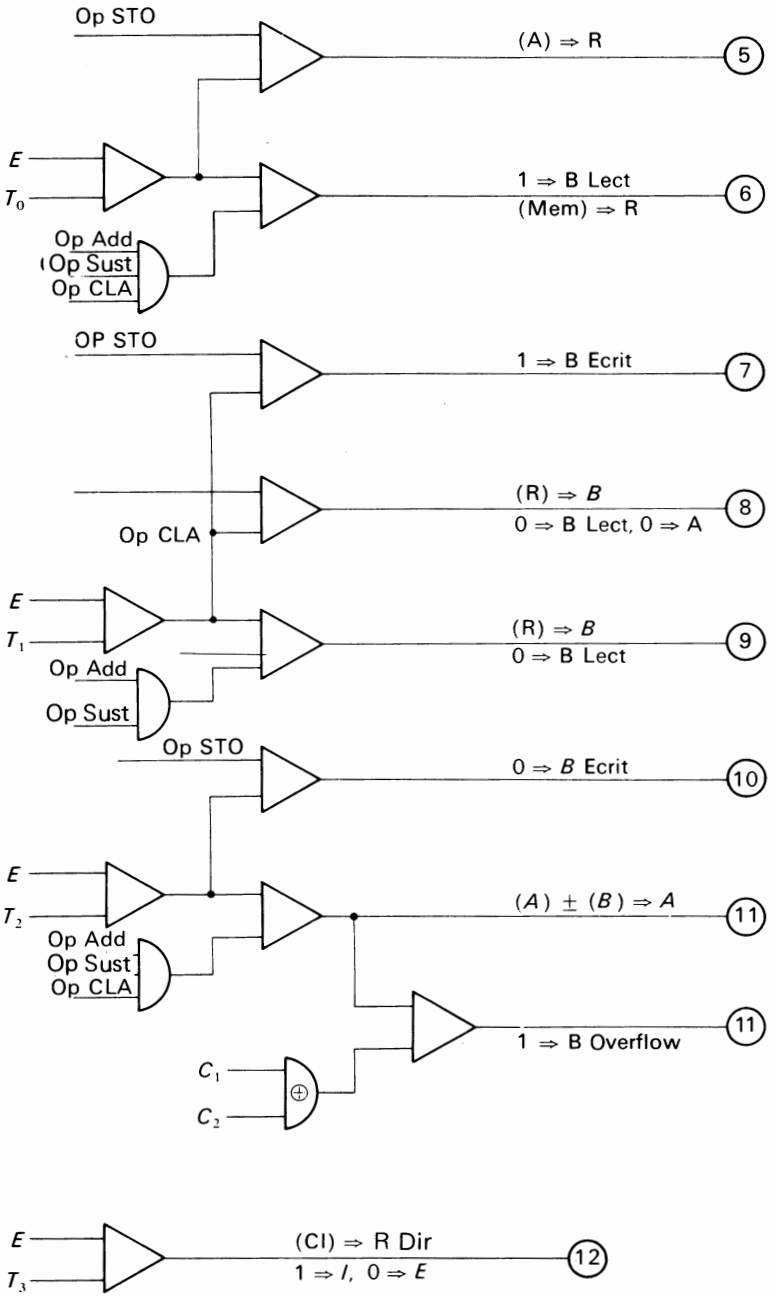


Fig. 18. *Ciclo de ejecución*

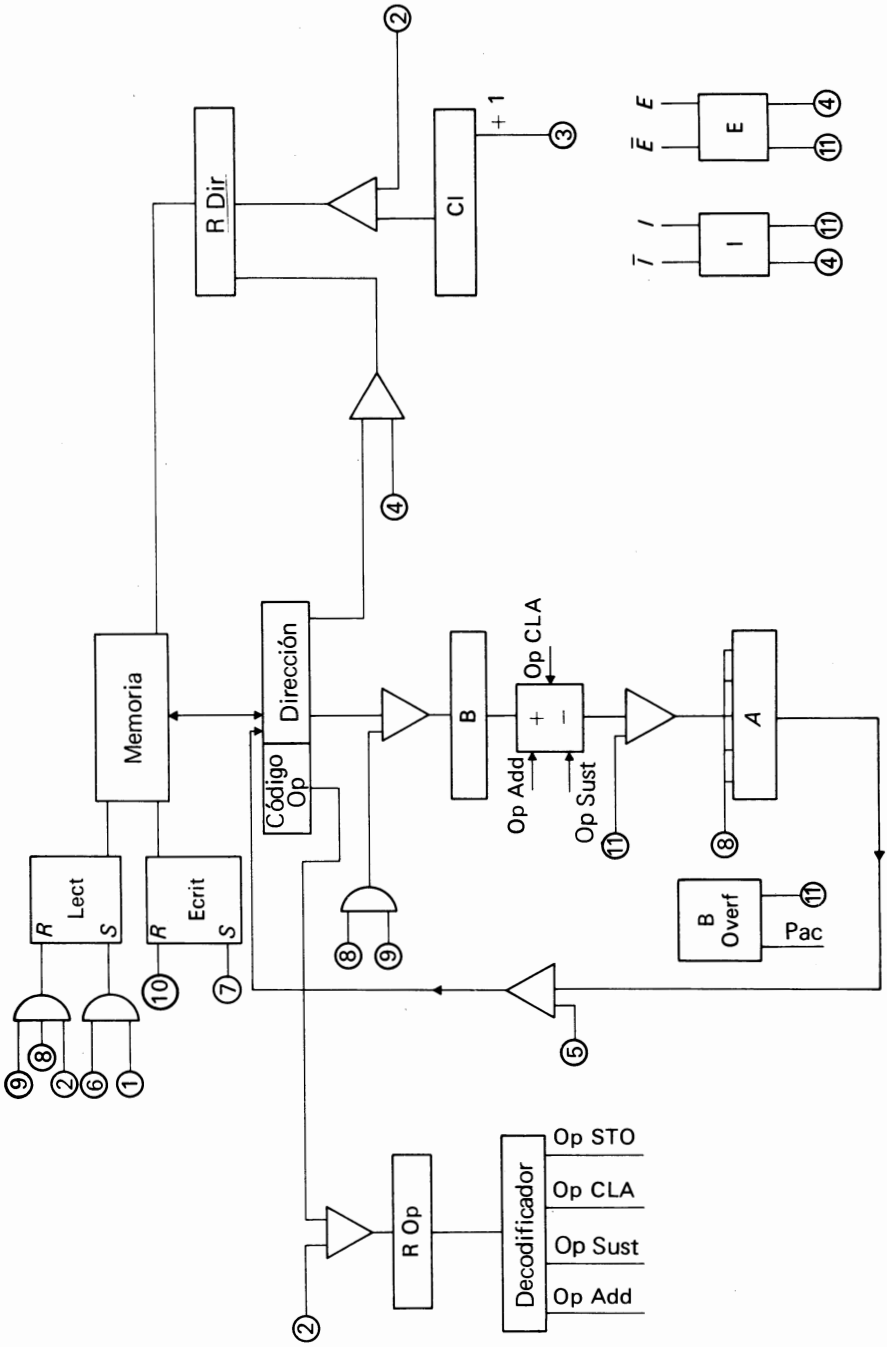


Fig. 19. *Esquema general del calculador estudiado en el texto*

7.1 Realización de las operaciones de bifurcación

Anteriormente hemos visto en el estudio de las operaciones de control que frecuentemente era necesario realizar una bifurcación en el programa. Esta bifurcación puede ser condicional o incondicional. En los dos casos, es necesario interrumpir la secuencia normal de las instrucciones. Cuando se trata de una bifurcación condicional ésta se realizará si están presentes ciertas condiciones (contenido del acumulador, cero, par, impar, negativo, rebasamiento de capacidad, etc.). Examinemos el caso de una bifurcación del programa si el contenido del acumulador es negativo. La instrucción de bifurcación es una instrucción particular que tiene un código de operación y una parte dirección que contiene la dirección de la instrucción siguiente a realizar (que es diferente de la instrucción que le sigue físicamente). Si la condición de bifurcación está presente se "bifurcará" efectivamente a esta instrucción. En caso contrario se continúa en secuencia. La figura 20 detalla las micro-operaciones realizadas por la operación de bifurcación.

Tiempos elementales	Micro-operaciones	Detalle de las micro-operaciones
IT_0	$1 \Rightarrow$ B Lect (Mem $\Rightarrow$ R)	Lectura de la instrucción. La puesta en 1 de la báscula Lect transfiere la instrucción al registro de memoria R.
IT_1	(Op) R $\Rightarrow$ R Op	El código de operación de la instrucción es transferido al registro de operación. La báscula Lect es puesta a cero.
$IT_2\ \overline{A_0}$	(CI) + 1 $\Rightarrow$ CI	El contenido del contador de instrucción aumenta en una unidad para preparar la lectura de la instrucción siguiente si $A_0 = 0$.
$IT_2\ A_0$	(Dir) R $\Rightarrow$ CI	La parte dirección de la instrucción es transferida al contador de instrucciones si $A_0 = 1$.
IT_3	(CI) $\Rightarrow$ R Dir	El contenido del contador de instrucciones es transferido al registro de direcciones. Este contiene la dirección de la instrucción siguiente en secuencia si $A_0 = 0$, o la nueva dirección de la instrucción después de la ruptura de la secuencia si $A_0 = 1$. No hay ciclo de ejecución.

Fig. 20. *Operación de bifurcación si A <0*

Las dos primeras micro-operaciones son idénticas a las del caso precedente: lectura de la instrucción y decodificación del código de operación. Seguidamente si $A_0 = 0$, es decir si el contenido del acumulador es positivo se aumenta el contenido de las instrucciones en una unidad. Si $A_0 = 1$, la parte dirección de la instrucción es transferida al contador de instrucciones. En los dos casos, la última operación es la transferencia del contenido del contador de instrucciones al registro de direcciones. Este registro contiene entonces la dirección de la instrucción siguiente que no está en secuencia con la anterior, sino que es una dirección cualquiera en el programa. Obsérvese que no hay ciclo de ejecución para la instrucción de bifurcación tal como la hemos realizado. En la figura 21 se representa el diagrama del ciclo de instrucción del calculador precedente al que se le ha añadido la operación de bifurcación.

7.2 Realización de las operaciones de desplazamiento

Anteriormente hemos visto que la operación de desplazamiento tiene un código de operación particular y que el número de desplazamientos a efectuar se indica en la parte dirección de la instrucción. Evidentemente existe un código de operación diferente para los desplazamientos a la derecha y a la izquierda. Los desplazamientos pueden interesar a varios registros de la máquina.

Para cada uno de estos registros tendremos dos códigos de operación diferentes según que se trate de un desplazamiento a la derecha o a la izquierda. El número de desplazamientos es introducido en un contador que disminuye en una unidad por cada desplazamiento. Cuando su contenido es cero termina la operación. La figura 22 representa el detalle de las micro-operaciones del ciclo de instrucción. Obsérvese que las dos primeras micro-operaciones son idénticas a las vistas anteriormente. La tercera prepara la lectura de la instrucción siguiente y transfiere el número de desplazamientos contenidos en la parte dirección de la instrucción al contador de desplazamientos *C*.

La cuarta micro-operación pone en 1 una báscula "Desplazamientos" y pone a cero la báscula *I*. La máquina ha terminado su ciclo de instrucción y no hay ciclo de ejecución. *Durante el tiempo necesario al desplazamiento, se detiene el distribuidor de tiempos.* Esto se obtiene por medio de la salida complementaria de la báscula "desplazamientos". Cuando el contador de desplazamientos está en cero, pone a cero la báscula "desplazamientos" y en 1 la báscula *I*. De este modo la máquina puede comenzar un nuevo ciclo *I*. La figura 23 representa el esquema de realización del desplazamiento a la derecha del contenido del acumulador. Nótese que el ciclo de instrucción es válido cualquiera que sea la naturaleza de la operación de desplazamiento a realizar, lo que es puesto de relieve por el mando Σ Op Dpz aplicado en los tiempos T_2 y T_3. Según el código de operación, la operación de desplazamiento se verificará sobre el acumulador, el registro Q o sobre los dos registros eventualmente, la selección es efectuada por el decodificador de operación que autoriza el desplazamiento de cada registro. En el caso de la figura 23, el acumulador es sometido a un desplazamiento a la derecha. La

figura 24 representa una extensión del ciclo de instrucción a las operaciones de desplazamientos. Cuando el calculador deba realizar operaciones más complejas tales como la multiplicación y la división, es necesario prever frecuentemente varios ciclos de instrucción y de ejecución.

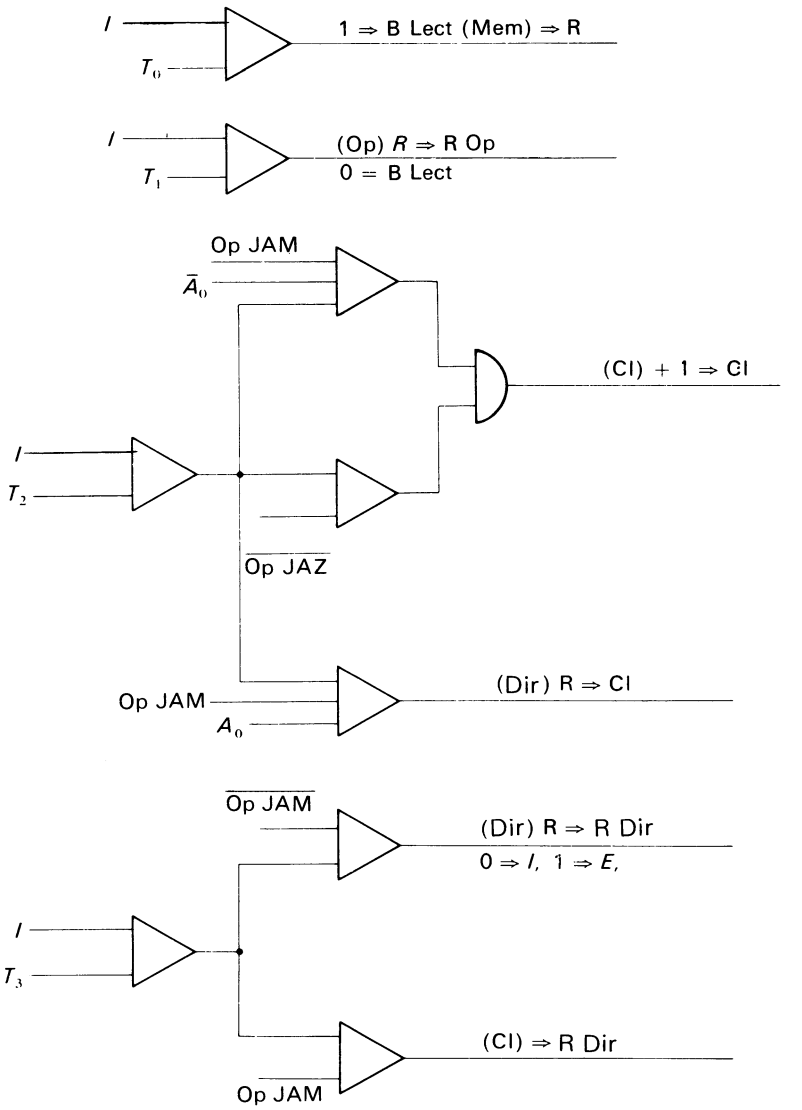


Fig. 21. *Extensión del ciclo de operación de la figura 16 para incluir la operación de bifurcación Op JAZ*

Tiempos elementales	Micro - operaciones	Detalle de las micro - operaciones
IT_0	1 ⇒ B Lect (Mem) ⇒ R	Lectura de la instrucción. La puesta en 1 de la báscula Lect transfiere la instrucción al registro de memoria R.
IT_1	(Op) R ⇒ R Op 0 ⇒ B Lect	El código de la operación de la instrucción es transferido al registro de operación. La báscula Lect es puesta a cero.
IT_2	(CI) + 1 ⇒ CI (Dir) R ⇒ C	El contenido del contador de instrucciones aumenta en una unidad para preparar la lectura de la instrucción siguiente y el contenido de la parte dirección de la instrucción es transferido al contador.
IT_3	0 ⇒ I, 1 ⇒ B Dpz (CI) ⇒ R Dir	La báscula I es puesta a cero. La báscula "Desplazamientos" es puesta en 1. La máquina ha terminado su ciclo de instrucción y no hay ciclo de ejecución.

Fig. 22. *Operación de desplazamientos*

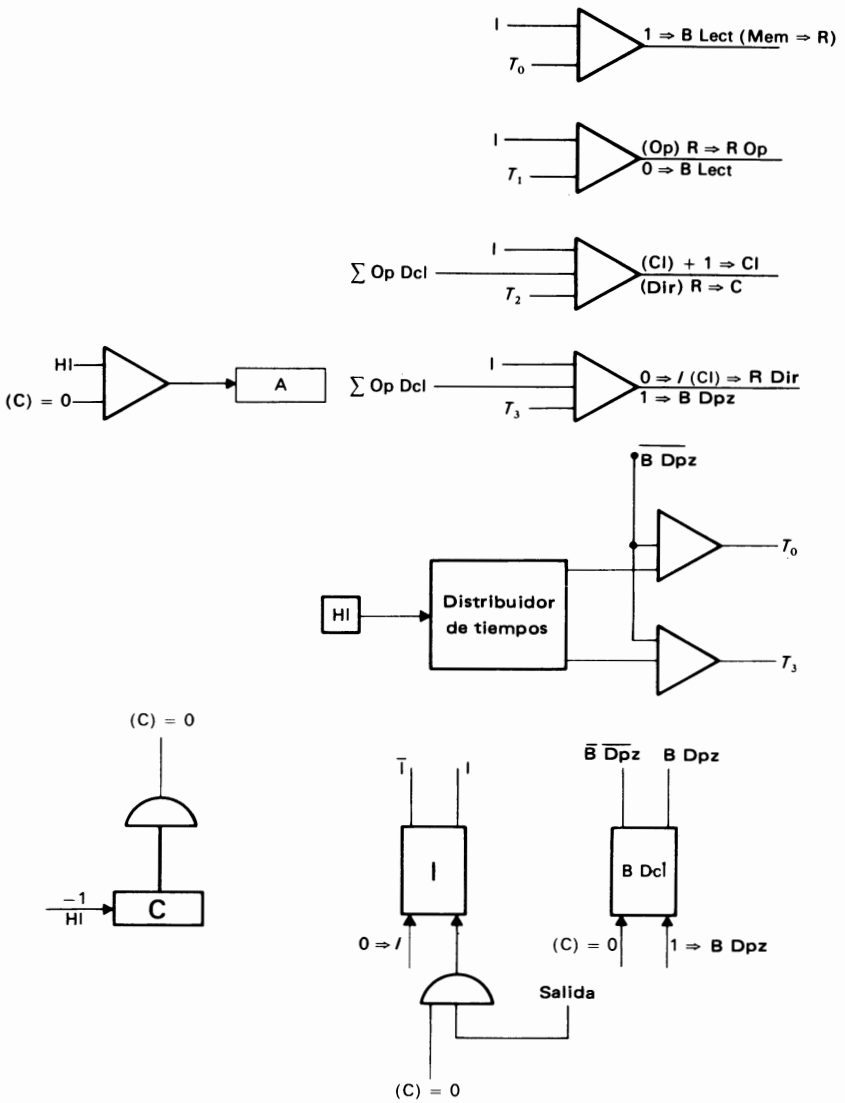


Fig. 23. *Realización de la operación de desplazamiento del contenido del acumulador*

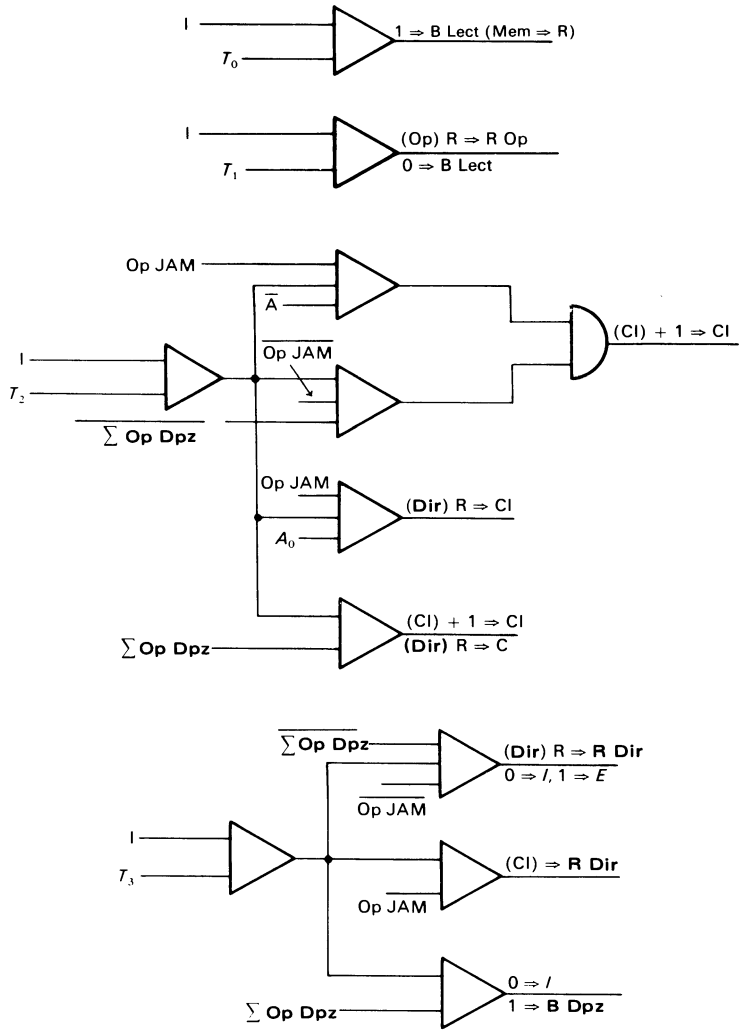


Fig. 24. *Extensión del ciclo de instrucción de la figura 20 a las operaciones de desplazamiento*

8. Estudio de un pequeño ordenador

En este párrafo nos proponemos estudiar un pequeño calculador de dos ciclos *I* y *E* que realiza las operaciones siguientes:

"Y"	0000
"O exclusivo"	0001
Desplazamiento a la derecha	0010
Desplazamiento a la izquierda	0011
Almacenamiento del acumulador en memoria *(STO)*	0100
Almacenamiento del contador de instrucciones en memoria	0101
Almacenamiento del contenido del índice en memoria *(STX)*	0110
Carga del acumulador *(LD)*	0111
Carga del registro de índice *(LDX)*	1000
Adición	1001
Sustracción	1010
Bifurcación incondicional	1011
Bifurcación si el contenido del acumulador es negativo	1100
Bifurcación si el contenido del acumulador es nulo	1101
Bifurcación si el contenido del registro de índice es nulo	1110
Espera	1111

o sea un total de 16 operaciones diferentes. La longitud de una palabra de memoria es de 11 bits. Se supone que la memoria interna del calculador está realizada con registros a básculas con el fin de evitar un ciclo de reescritura. Hemos visto que el desarrollo de los circuitos integrados permite un desarrollo de este tipo. El formato de una instrucción y de un dato corresponden a la configuración de la figura 25.

Nótese la posibilidad de utilizar un registro de índice *Ix*

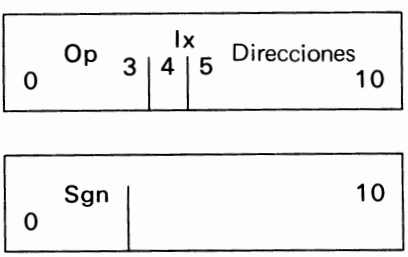


Fig. 25. *Formato de una instrucción y de un dato en el calculador estudiado en el texto*

La parte código de operación de la instrucción tiene cuatro posiciones, una posición *Ix* es reservada para la indexación, las posiciones de 5 a 10 son para la dirección del dato a tratar. Tendremos pues, 64 palabras de memoria con las direcciones sucesivas de 00 a 63. Para los números negativos se utilizará la representación en complemento a 2. Se han eliminado las operaciones de

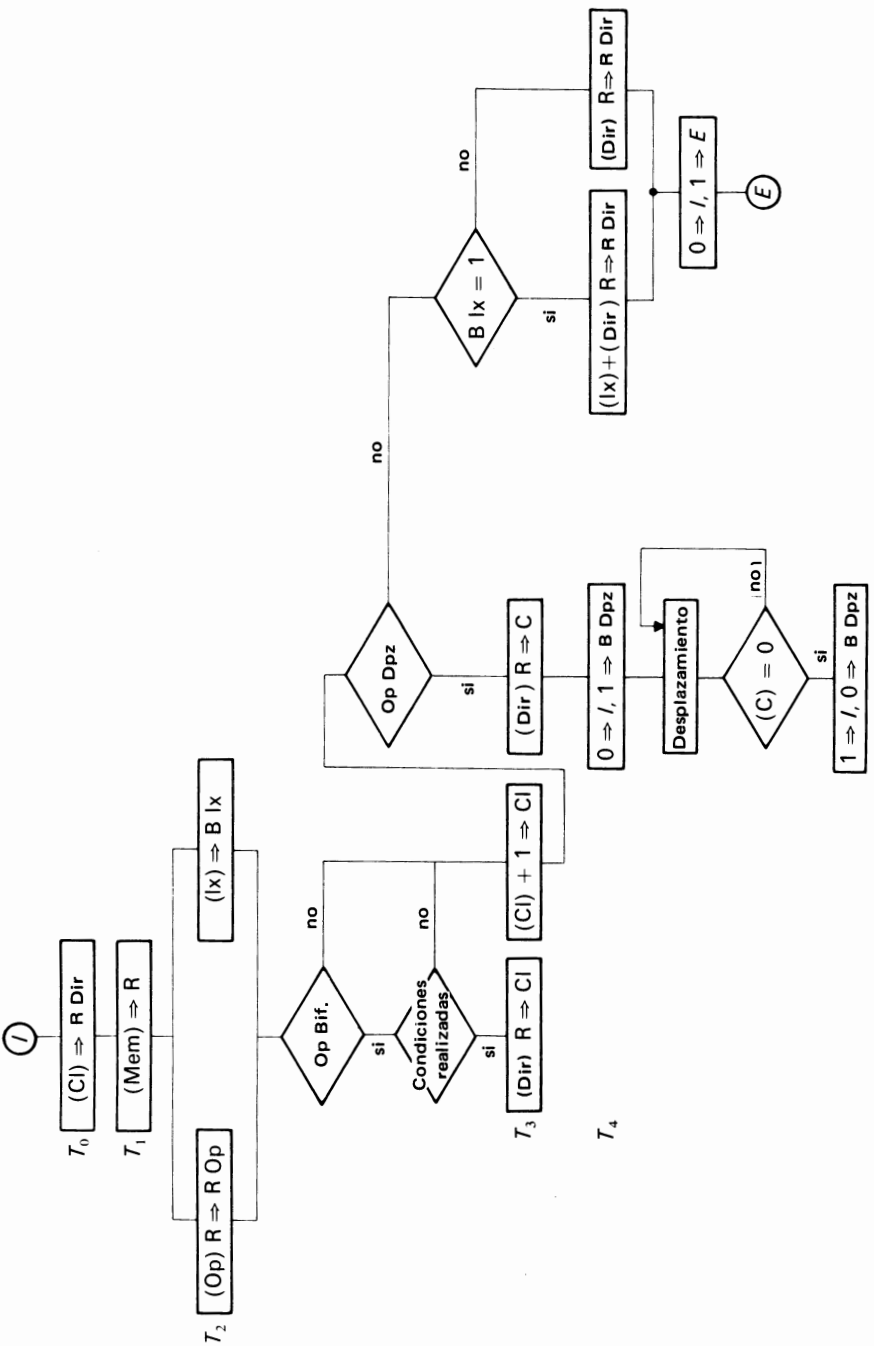


Fig. 26. *Ciclo I del calculador estudiado en el texto*

multiplicación y división para no complicar el estudio. Más adelante indicaremos cómo modificar los resultados para realizar estas dos operaciones. En la figura 12 se representa el esquema general del calculador. El adicionador-sustractor será del tipo estático. Con un mando del ordenador, éste proporciona la suma, la diferencia, la multiplicación lógica y la suma sin arrastre de dos números binarios situados en el registro de memoria y en el acumulador, el resultado aparece en el acumulador. Un adicionador-sustractor de este tipo ha sido estudiado en el tomo primero. El acumulador registra los resultados elaborados por el adicionador-sustractor pudiendo desplazar su contenido, a una orden del ordenador, de un cierto número de posiciones a la derecha o a la izquierda. En el caso de un desplazamiento a la derecha se conservará el signo. En el capítulo *I* se estudió un acumulador de este tipo. La decodificación de la instrucción durante el ciclo *I* se efectúa según el organigrama de la figura 26.

Obsérvese la existencia de una báscula *B Ix* destinada a memorizar la posición *Ix* del formato de la instrucción. Este ciclo *I* es ligeramente diferente al estudiado anteriormente. Si la instrucción no es indexada, su parte dirección se introduce sin modificar en el registro de dirección, si es indexada el contenido del registro de índices es añadido a la dirección. La decodificación de la instrucción se efectuará en cinco tiempos elementales de T_0 a T_4, lo mismo ocurre para la ejecución de todas las operaciones. El ciclo *I* está representado en la figura 26.

La realización del ciclo *I* corresponde a la figura 27, las tres primeras micro-operaciones IT_0 IT_1, IT_2 son las siguientes: transferencia del contenido del contador de instrucciones al registro de instrucciones, transferencia del contenido de la memoria a la dirección indicada por el registro de direcciones, al registro de memoria, transferencia de la parte operación y del índice del registro de memoria, respectivamente, al registro de operación y a la báscula *B Ix*. En el tiempo T_3 se examinan los casos en los cuales las operaciones de bifurcación son o no realizadas. No se ejecutarán si estamos en presencia de una operación de bifurcación (Σ Op Bf), o si estamos en presencia de una operación de bifurcación en la que sus condiciones están ausentes. Todas estas condiciones son agrupadas en 4. En estos diferentes casos se incrementa en una unidad el contador de instrucciones para preparar la lectura en secuencia de la instrucción siguiente. Las operaciones de bifurcación se realizarán cuando las condiciones de bifurcación estén presentes. Estos casos son agrupados en 5. Entonces se introduce la parte dirección del registro que contiene la dirección de bifurcación, en el contador de instrucciones. Esta nueva dirección es introducida en el registro de direcciones al principio del siguiente ciclo *I*. No hay ciclo de ejecución. También en IT_3, si se trata de una operación de desplazamientos a la derecha o a la izquierda (Σ Op Dpz) la parte dirección del registro de memoria, que contiene el número de desplazamientos a efectuar, será introducido en un contador de desplazamientos *C* cuya función es muy importante en todos los ordenadores. Al final del ciclo *I* en el tiempo T_4 para todas las operaciones excepto las de bifurcación y desplazamiento se realizará el ciclo de ejecución ($0 \Rightarrow I$, $1 \Rightarrow E$).

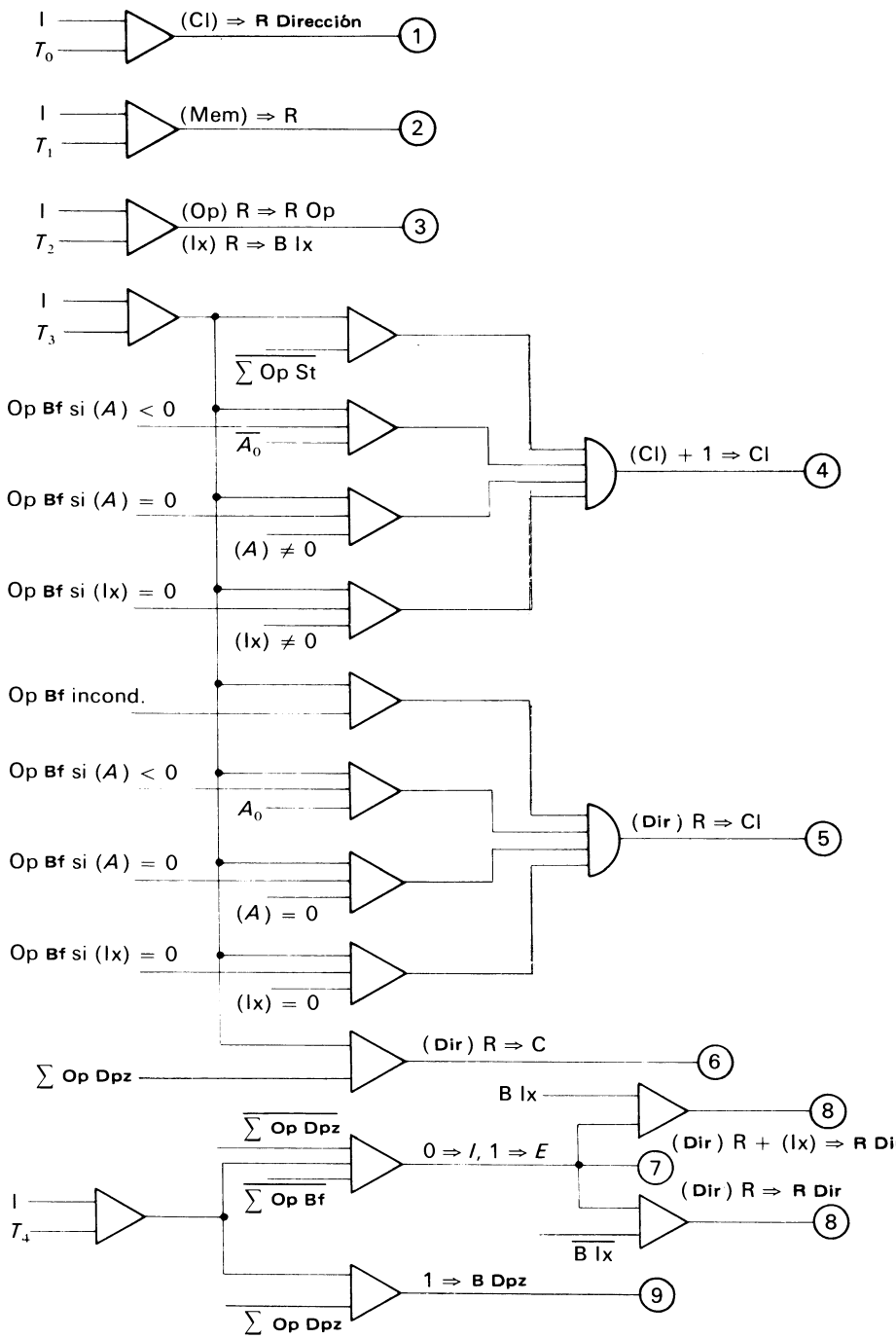


Fig. 27. *Realización del Ciclo I del calculador estudiado en el texto*

Entonces se introduce el contenido de la parte dirección del registro de memoria, modificado o no por el contenido de *B Ix*, en el registro de dirección. En las operaciones de desplazamiento una báscula B Dpz es puesta en 1 memorizando la operación hasta que el contenido de C sea nulo. La salida falsa de la báscula B Dpz suprime los impulsos de reloj que no sean aplicados simultáneamente al acumulador y al contador de desplazamientos *C*. Cuando el contenido de *C* es nulo se pone a cero la báscula B Dpz que permite entonces el paso de los impulsos de reloj hacia el distribuidor de tiempos, el calculador comienza un nuevo ciclo *I* en T_0, figura 29. Al final de los desplazamientos se pasa inmediatamente a IT_0, lo que no es obligatorio si la operación de desplazamiento es ejecutada durante un ciclo *E*. En efecto, el número de desplazamientos puede ser superior o inferior al número de tiempos elementales del ciclo *E*. En este caso es necesario continuar hasta el fin del ciclo antes de comenzar un nuevo ciclo *I*, lo que ocasiona una pérdida de tiempo. Se evita este inconveniente utilizando el método anterior. Examinemos ahora el ciclo de ejecución *E* de este calculador. La figura 28 representa los ciclos de ejecución para todas las operaciones excepto las de bifurcación y desplazamientos. El mismo distribuidor de tiempos es utilizado para el ciclo *E*. Este comprende, pues, el mismo número de micro-operaciones que el ciclo *I*, la vuelta al ciclo *I* se efectúa también al final del ciclo *E* en el tiempo T_4. Todas las operaciones del ciclo *E* se efectúan en dos tiempos elementales T_0 y T_1, los tiempos T_2 y T_3 no son utilizados, el tiempo T_4 es para el retorno al ciclo *I*. En el caso de la operación "espera", la parte dirección de la instrucción no es

Tiempos elementales	Cambio de A	Cambio de Ix	Almacenamiento de A	Almacenamiento de Ix	Almacenamiento de CI
ET_0	(Mem) ⇒ R	(Mem) ⇒ R	(A) ⇒ R	(Ix) ⇒ R	(CI) ⇒ R
ET_1	(R) ⇒ A	(R) ⇒ Ix	(R) ⇒ Mem	(R) ⇒ Mem	(R) ⇒ Mem
ET_2					
ET_3					
ET_4	0 ⇒ *E*, 1 ⇒ *I*	0 ⇒ *E*, 1 ⇒ *I*	0 ⇒ *E*, 1 ⇒ *I*	0 ⇒ *E*, 1 ⇒ *I*	0 ⇒ *E*, 1 ⇒ *I*
Tiempos	Adición	Sustracción	"Y"	⊕	Espera
ET_0	(Mem) ⇒ R	(Mem) ⇒ R	(Mem) ⇒ R	(Mem) ⇒ R	(Mem) ⇒ R
ET_1	(A) + (R) ⇒ A $C_1 \oplus C_2$ ⇒ B Over	(A) − (R) ⇒ A $C_1 \oplus C_2$ ⇒ B Over	(A) (R) ⇒ A	(A) + (R) ⇒ A	
ET^2					
ET_3					1 ⇒ B Espera
ET_4	0 ⇒ *E*, 1 ⇒ *I*	0 ⇒ *E*, 1 ⇒ *I*	0 ⇒ *E*, 1 ⇒ *I*	0 ⇒ *E*, 1 ⇒ *I*	Arranque por pupitre 1 ⇒ *B* Espera (0 ⇒ *E*, 1 ⇒ *I*)

Fig. 28. *Ciclo de ejecución del calculador estudiado en el texto*

utilizada. No obstante se efectúa la transferencia del contenido de la memoria al registro de memoria R, a pesar de que esta operación sea inútil. En ET_3 se para el calculador suprimiendo los impulsos de reloj por medio de una báscula B. Esta espera es puesta en 1 en el tiempo T_3. El calculador se volverá a poner en marcha apoyando sobre el botón "Arranque" colocado en el pupitre, el cual efectúa manualmente la operación, esto encadena el siguiente ciclo I en el tiempo ET_4. También tenemos la posibilidad de parar el calculador por medio de un botón "Parada" colocado en el pupitre de mando. Este botón "Parada" controla a una báscula Parada cuya salida verdad está conectada a la entrada verdad de la báscula Espera de tal modo que la parada del calculador se efectúa siempre en ET_3. Puede referirse el lector a la salida 20 de la figura 30 que representa la realización del ciclo de ejecución del calculador. Obsérvese que las operaciones adición,

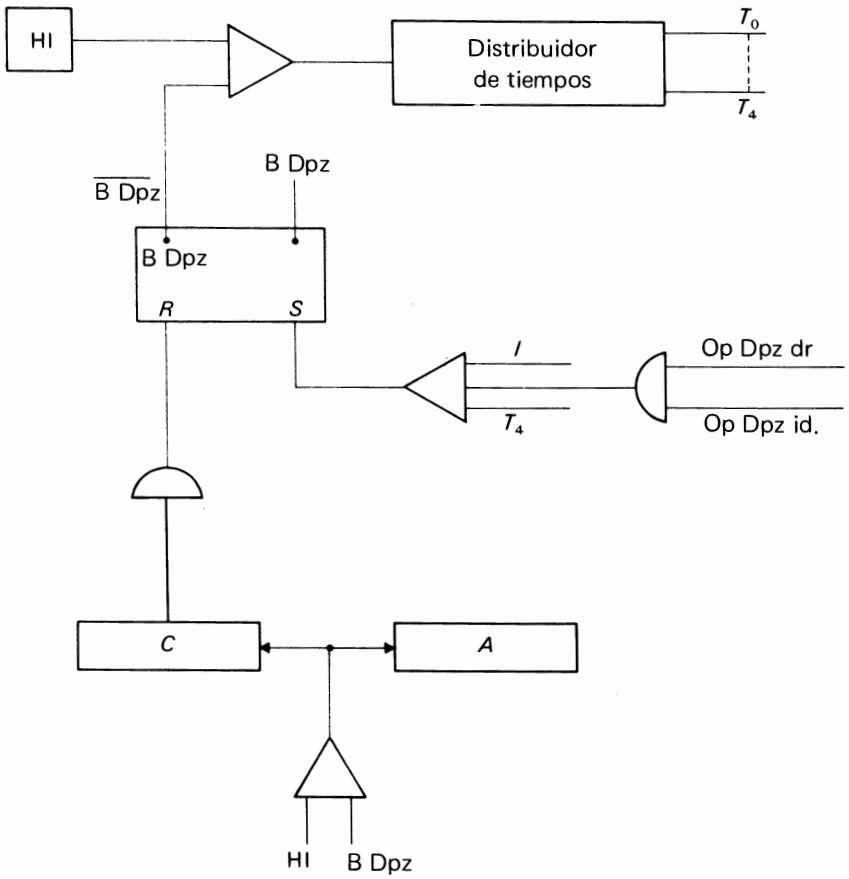


Fig. 29. *Realización de las operaciones de desplazamiento*

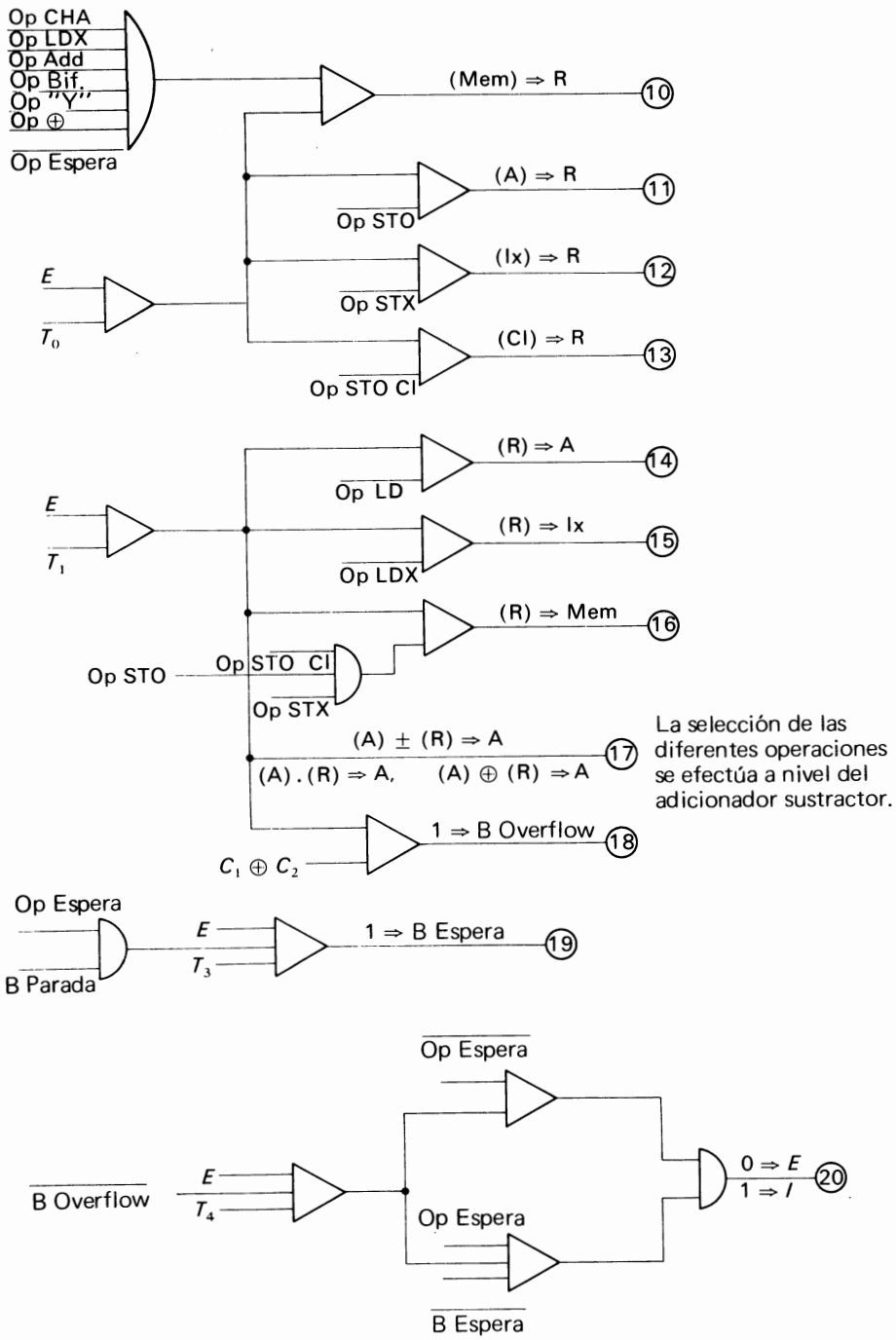


Fig. 30. *Realización del ciclo de ejecución del calculador estudiado en el texto*

sustracción "Y" y ⊕ no están condicionadas en el ciclo *E*. La selección entre estas operaciones se efectúa, como hemos visto, a nivel de adicionador-sustractor. Estas operaciones se realizan en ET_1 al igual que la puesta en 1 de la báscula B Overflow, si las condiciones de rebasamiento del acumulador están presentes. En este caso el ordenador se para al final de la adición, no obstante el distribuidor de tiempos continúa funcionando: los impulsos T_0 a T_4 son suprimidos por la báscula B Overflow. La figura 31 representa las condiciones de parada del distribuidor de tiempos. El encadenamiento de los ciclos de instrucción y de ejecución se efectúa según la figura 32. Nótese que se abandona el ciclo *I* ($0 \Rightarrow I$) en el tiempo T_4 a condición de que no se trate de una operación de desplazamientos o de bifurcación. Se reúnen en la entrada *R* de la báscula *E* todas las condiciones de fin de operación (20) (Fig. 30). Las figuras 33 y 34 representan la organización general del calculador en los dos ciclos de funcionamiento *I* y *E*.

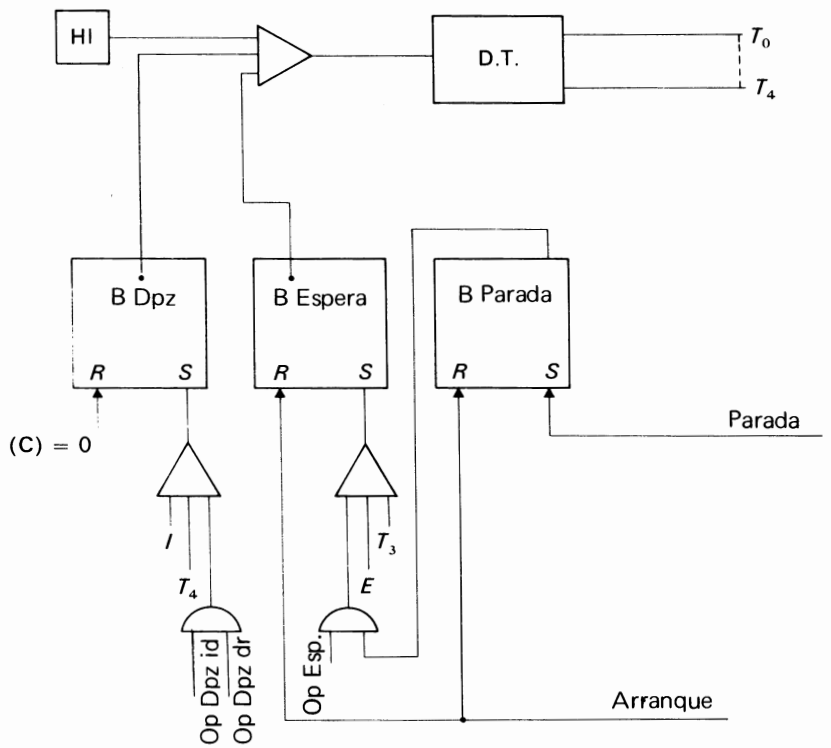


Fig. 31. *Condiciones de parada del distribuidor de tiempos*

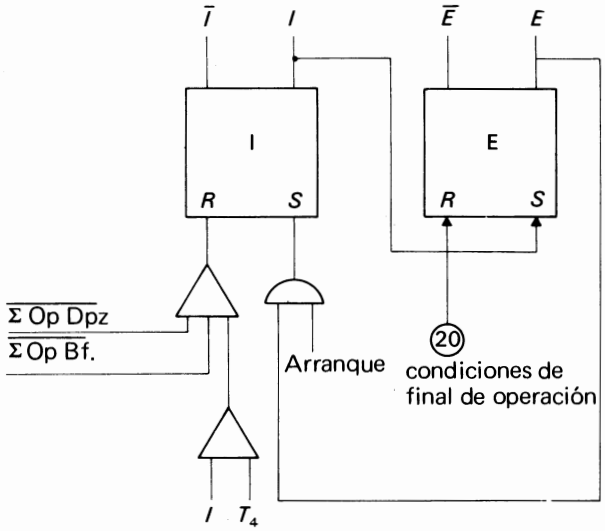


Fig. 32. *Encadenamiento de los ciclos I y E*

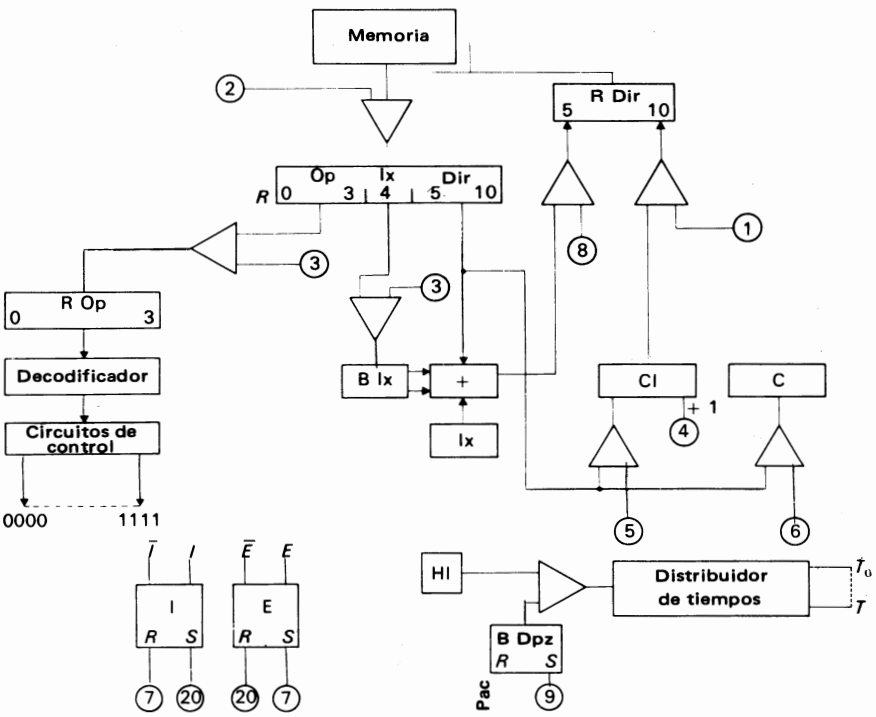


Fig. 33. *Organización general del ordenador estudiado en el texto (C) =0 durante el ciclo de instrucción*

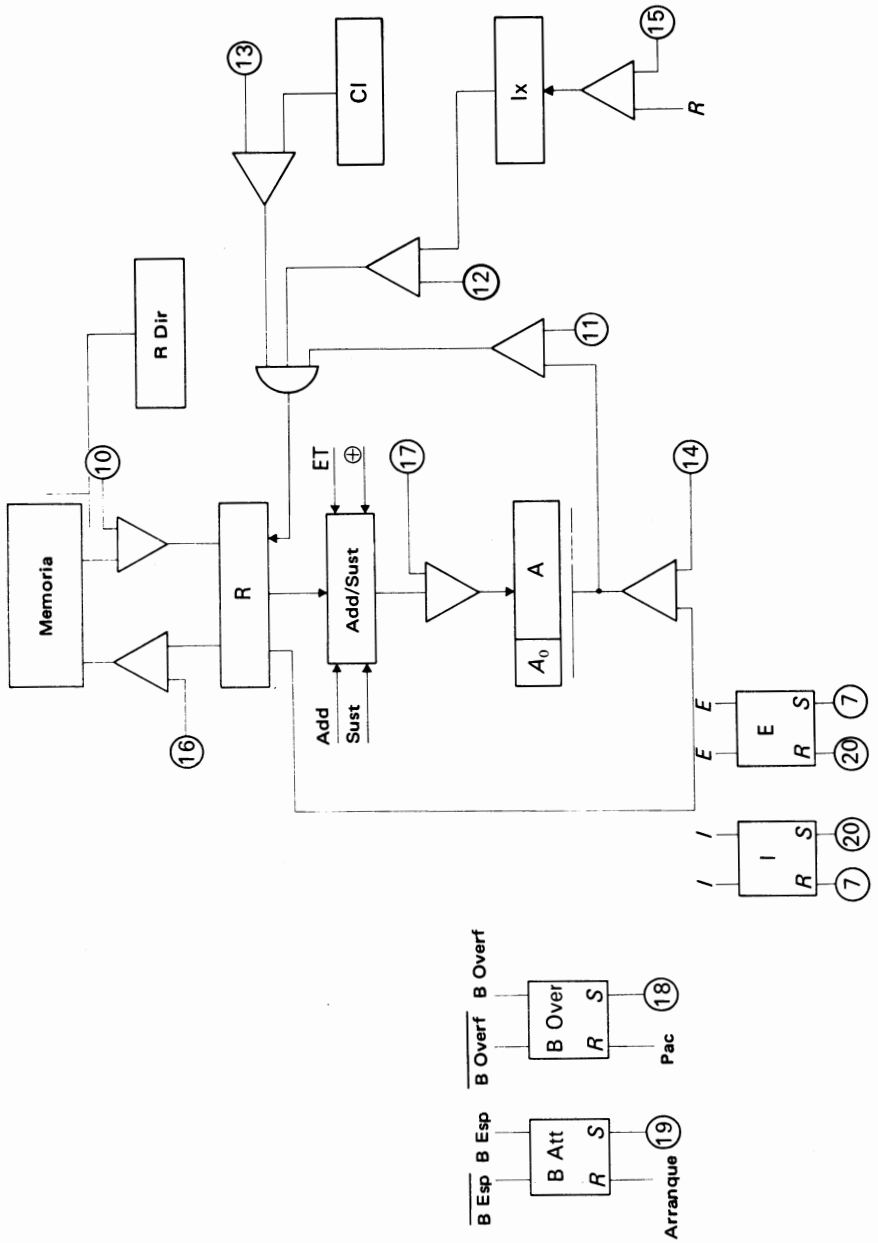


Fig. 34. *Organización general del ordenador estudiado en el texto durante el ciclo de ejecución*

9. Realización de los diferentes elementos

Todos los elementos que componen un calculador:
- registros,
- contador de instrucciones y contador de desplazamientos,
- acumulador,
- adicionador-sustractor,
- decodificadores,

han sido estudiados detalladamente en los tomos I y II. En particular se encontrará en este volumen el estudio del adicionador-sustractor y del acumulador utilizados en este calculador. A título de ejemplo estableceremos el esquema del contador de instrucciones. Según lo que hemos visto anteriormente el contador de instrucciones incrementa su contenido en una unidad en el tiempo T_2 del ciclo I si las condiciones de bifurcación no se cumplen así como para las demás operaciones. Por otra parte recibe el contenido de la parte dirección del registro de memoria R en el mismo instante del ciclo I si se trata de una operación de bifurcación incondicional o de una operación de bifurcación en la que se cumplen sus condiciones. Se utilizarán las básculas R-S-T. La figura 35 representa el esquema para los tres últimos pesos. Obsérvese la existencia de una puesta a cero del contador que se efectuará al principio del desarrollo del programa. Para la

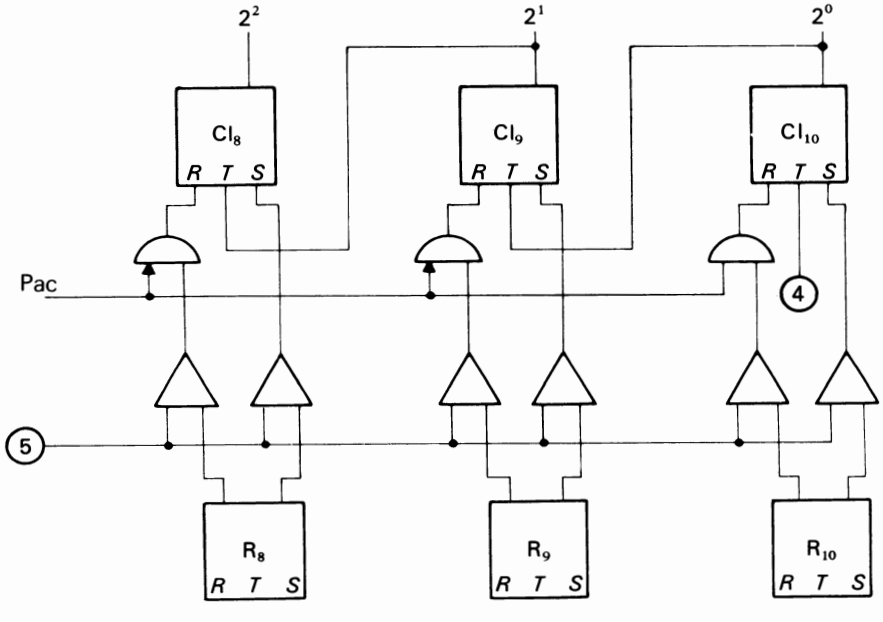


Fig. 35. *Realización del contador de instrucciones*

realización de los demás subconjuntos del calculador se procederá del mismo modo haciendo referencia al capítulo I.

9.1 Extensión del código de operación

Si se desea ampliar el código de operación a las operaciones de multiplicación y división será necesario prever una posición suplementaria en la parte reservada al código de operación de la instrucción, lo que equivale a disminuir el número de datos susceptibles de ser almacenados en memoria si se conserva la misma longitud de palabras. Para estas operaciones se utilizará el algoritmo de Booth para la multiplicación y el método de comparación de los signos del divisor y de los restos parciales para la división, si los números negativos son representados por sus complementos a 2. Otro método consiste en representar los números en valor absoluto y con signo únicamente para la multiplicación y la división. Se utiliza el método clásico para la multiplicación y el método de restablecimiento de restos para la división. Naturalmente el resultado deberá transformarse en complemento a 2 si es negativo, lo que necesita un dispositivo adicional mandado por el primer peso A_1 del acumulador. Por otra parte según el método utilizado, deberá modificarse el distribuidor de tiempos que habrá de tener tantos tiempos elementales T_0 a T_n como micro-operaciones haya que realizar en la operación más larga (división).

10. Tendencias en la concepción de los ordenadores modernos

La tendencia en los ordenadores a palabras recientes es la utilización de máquinas de dirección simple. El formato de las instrucciones y de los datos, puede, según los casos, componerse de dos palabras consecutivas. Se habla de una instrucción en formato largo y de datos en doble precisión en contraposición a una instrucción en formato corto y a un dato en simple precisión que solamente se componen de una palabra. El ordenador contendrá uno o varios ciclos de instrucción sucesivos según que la instrucción sea de formato corto o de formato largo, que utilice la modificación de dirección por un índice, o que sea direccionada directamente o indirectamente. Las instrucciones de bifurcación son agrupadas en una sola instrucción, indicando las diferentes condiciones de bifurcación en el formato de la instrucción. Las instrucciones que tratan los datos en doble precisión así como ciertas operaciones aritméticas y de entradas-salidas serán realizadas en varios ciclos de ejecución. El número de operaciones elementales en cada ciclo no corresponde a la operación más larga. Para ésta y para algunas otras no se tomará el ciclo en su comienzo, pero se procederá a su extensión manteniendo la báscula I o E sobre el ciclo en ejecución y continuando las operaciones elementales del ciclo por medio de los impulsos de reloj.

Anexo

Complementos al capítulo 4

1. Rebasamiento de capacidad en la adición y la sustracción

El rebasamiento de capacidad del acumulador en la adición y la sustracción puede ser verificado igualmente una vez terminada la operación comparando el signo de los dos operandos en una báscula B Sgn, el signo del resultado A_0 y el arrastre más allá del signo (B Arrastre). El lector puede verificar fácilmente la validez de la figura A1 con algunos ejemplos simples.

B Sgn	A_0	B Arrastre	Rebasamiento de capacidad
0	0	0	no
0	0	1	si
0	1	0	si
0	1	1	no
1	0	0	si
1	0	1	no
1	1	0	no
1	1	1	si

Fig. A1. *Condiciones de rebasamiento de capacidad en la adición y sustracción.*

2. Las operaciones de desplazamientos en el multiplicador de Booth

2.1 Multiplicador de Booth primera versión

Según lo que hemos visto en el capítulo 4, las operaciones de desplazamiento no precedidas de una operación de adición o de sustracción ($\overline{Q_5 \oplus Q_s} = 1$) se efectúan siempre conservando el signo del acumulador. Por el contrario las operaciones de desplazamiento asociadas a una operación de adición o de sustracción ($Q_5 \oplus Q_s = 1$), se efectúan invirtiendo el signo del acumulador en el caso de un rebasamiento de capacidad. En este último caso se determinará el signo del acumulador comparando el signo del multiplicando con la operación que se acaba de efectuar. El signo del acumulador será positivo si la operación que se acaba de efectuar es una adición y que el multiplicando sea positivo, o que la operación sea una sustracción y el multiplicando negativo. Será negativo si la operación que se acaba de efectuar es una adición y que el multiplicando sea negativo o que la operación sea una sustracción y el multiplicando positivo. El lector puede remitirse al tomo 1. La báscu-

la signo del acumulador comprenderá dos vías: una sensibilizada por $\overline{Q_5 \oplus Q_s}$ que mantiene el signo en los desplazamientos, otra por $Q_5 \oplus Q_s$ que impone el signo conveniente a la báscula signo, es decir, según lo que acabamos de decir, $\overline{Q_5}\ Q_s R_1 + \bar{Q}_5\ Q_s \bar{R}_1$.

2.2 Multiplicador de Booth segunda versión

Como para la primera versión la báscula signo del acumulador tiene dos vías: una sensibilizada por B Op $\oplus$ Q_5 que mantiene el signo durante los desplazamientos, otra sensibilizada por $\overline{\text{B Op} \oplus Q_5}$ impone a la báscula el signo conveniente, es decir $\overline{\text{B Op}} \oplus R_1$. Recuérdese que la báscula B Op está en 1 en el caso de una adición y sobre 0 en el caso de una sustracción.

3. División de — 144 por + 12

Este ejemplo está destinado a ilustrar la observación hecha en la página 186, capítulo 4, relativo a la detección de un resto idénticamente nulo durante la división.

B resto	Acumulador	Registro Q	C	Observaciones
0	(1) 1 0 1 1	1 0 0 0 0	5	Principio de la división
	1 0 1 1 1	0 0 0 0 0	4	Dpz id
	+ 0 1 1 0 0			Adición
	(0) 0 0 1 1	0 0 0 0 $\underline{1}$		$1 \Rightarrow Q_5$
	0 0 1 1 0	0 0 0 $\underline{1}$ 0	3	Dpz id
	− 0 1 1 0 0			Sustracción
	1 1 0 1 0	0 0 0 $\underline{1}$ $\underline{0}$		$0 \Rightarrow Q_5$
	1 0 1 0 0	0 0 $\underline{1}$ $\underline{0}$ 0	2	Dpz id
	+ 0 1 1 0 0			Adición
1	(0) 0 0 0 0	0 0 $\underline{1}$ 0 $\underline{1}$		$1 \Rightarrow Q_5$
	0 0 0 0 0	0 $\underline{1}$ 0 $\underline{1}$ 0	1	Dpz id
	− 0 1 1 0 0			Sustracción
	(1) 0 1 0 0	0 $\underline{1}$ 0 $\underline{1}$ $\underline{0}$		$0 \Rightarrow Q_5$
	0 1 0 0 0	$\underline{1}$ 0 $\underline{1}$ 0 0	0	Dpz id
	+ 0 1 1 0 0			Adición
1	(1) 0 1 0 0	$\underline{1}$ 0 $\underline{1}$ 0 0		$0 \Rightarrow Q_5$
	0 0 0 0 0			B Resto = 1
	Resto = 0	Cociente = − 12		Puesta a cero del resto
				Dividendo negativo, divisor positivo y resto nulo: cociente en valor real

Fig. A2. Detalle de la división de −144 por +12, obsérvese el resto parcial nulo después de la tercera operación que pone en 1 la báscula B resto.

Los restantes dígitos del dividendo son 00 indicando un resto final nulo (B Resto = 1).

Bibliografía

Obras generales

AIKEN, *Synthesis of Electronic Computing and Control Circuits*, Harvard Press.

BOOLE, G., *The Mathematical Analysis of Logic*. McMillan, 1848. Nouvelle édition : Blackwell, Oxford, 1948.

BOOLE, G., *An Investigation of the Laws of Thought*. Londres, 1854. Nouvelle édition, Chicago, 1916.

BARON, R., PICCINILLI, A., *Digital Logic and Computer Operations*, 1967.

BOOTH, D. and BOOTH, V., *Automatic Digital Calculators*. Butterworths, 1953.

MCCLUSKEY, *Introduction to the Theory of Switching Circuits*. McGraw Hill, 1965.

CHU, *Digital Computer Design Fundamentals*. McGraw Hill, 1962.

DEBRAINE, P., *Machines de traitement de l'information*. Tomes 1 et 2, 1967 et 1969.

DENIS-PAPIN, M., KAUFMANN, A., FAURE, R., *Cours de calcul booléien appliqué*. Albin Michel, 1963.

HELLERMAN, H., *Digital Computer System Principles*, 1967.

KINTNER, P., *Electronic Digital Techniques*. McGraw Hill, 1968.

KUNTZMANN, J., *Algèbre de Boole*. Dunod, 1969.

LAGASSE, J., ERCEAU, J., *Logique combinatoire et séquentielle*. Dunod, 1969.

LAURET, A., *Principes de programmation des ordinateurs*. Masson, 1967.

PFISTER, *Logical Design of Digital Computers*. Wiley and Sons, 1958.

DU ROSCOAT, J., *Conception de la programmation des ordinateurs*. Masson, 1967.

SMITH, C. V. L., *Electronic Digital Computers*. McGraw Hill, 1959.

SOUBIES-CAMY, H., *Les techniques binaires et le traitement de l'information*. Dunod, 1966.

SPEISER, A., *Digitale Rechenanlagen*. Springer Verlag, 1965.

WEYH, U., *Elemente der Schaltungsalgebra*. R. Oldenborng, 1967.

Artículos de interés general

[1] SHANNON, C. E., « A Symbolic Analysis of Relay and Switching Circuits. » *Trans A.I.E.E.*, vol. 57, pp. 713-723, 1938.

[2] SHANNON, C. E., « The Synthesis of Two-terminal Switching Circuits. » *B.S.T.J.*, vol. 28, pp. 59-98. Janvier 1949.

[3] MCCLUSKEY, « Minimization of Boolean Functions. » *B.S.T.J.* Novembre 1956, pp. 1417-1444.

[4] McCLUSKEY, « Logical Design Theory of NOR Gate Networks with no complemented Inputs, Switching Circuit Theory and Logical Design. » *Proc. Fourth Ann. Symposium*, Chicago, 28-30 septembre 1963, pp. 105-116.

[5] KARNAUGH, M., « The Map Method for Synthesis of Combinational Logic Circuits. » *A.I.E.E. Commun. Electronics.* Novembre 1953, pp. 593-599.

[6] VEITCH, E. W., « A Chart Method for Simplifying Truth Functions. » *Proc. Computing Machinery Conf.* 2-3 mai 1952, pp. 593-599.

[7] QUINE, W. V., « A Way to Simplify Truth Functions. » *Am. Math. Monthly*, vol. 62, n° 9, pp. 627-631. Novembre 1955.

[8] BURKS, A. W., GOLDSTINE, H. and VON NEUMANN, J., « Preliminary Discussion of the Logical Design of an Electronic Computing Instrument. » The Institute for Advanced Study, Princeton, N.J., 1947.

[9] LEINER, A. L., SMITH, J. L. and WEINBERGER, A., « System Design of Digital Computer at the National Bureau of Standards. » Natl. Bur. of Standards, Circular 591. Février 1958.

[10] GILCHRIST, B., POMERENE, J. H. and WONG, S. Y., « Fast-carry logic for digital computers. » *IRE Trans. on Electronic Computers*, vol. EC-4, pp. 133-136. Décembre 1955.

[11] LEHMAN. M., « High-speed digital multiplication. » *IRE Trans. on Electronic Computers*, vol. EC-6, pp. 204-205. Septembre 1957.

[12] ROBERTSON, J. E., « A new class of digital division methods. » *IRE Trans. on Electronic Computers*, vol. EC-7, pp. 218-222. Septembre 1958.

[13] BLOCH, E., « The engineering design of the Stretch Computer. » *Proc. EJCC*, Boston, Mass., pp. 48-58. 1-3 décembre 1959.

[14] CAMPBELL, S. J. and ROSSER, G. H. Jr., « An Analysis of Carry Transmission in Computer Addition, » preprints of papers presented at the 13th Nat'l. Meeting of the ACM, Univ. of Illinois, Urbana. 11-13 juin 1958.

[15] BURTSEV, V. S., « Accelerating Multiplication and Division Operations in High-Speed Digital Computers. » Exact Mechanics and Computing Technique, Acad. Sci. USSR, Moscou, 1958.

[16] ROBERTSON, J. E., « Theory of Computer Arithmetic Employed in the Design of the New Computer at the University of Illinois. » Digital Computer Lab., University of Illinois, Urbana, file n° 319. Juin 1960.

[17] AVIZIENIS, A., « A Study of Redundant Number Representation for Parallel Digital Computers. » Digital Computer Lab., University of Illinois, Urbana, Rept. n° 101. 20 mai 1960.

[18] FRIEMAN, C. V., « A Note on Statistical Analysis of Arithmetic Operations in Digital Computers, » *P.I.R.E.*, janvier 1961, pp. 91-103.

[19] McSORLEY, « High-Speed Arithmetic in Binary Computers. » *P.I.R.E.*, janvier 1961, pp. 67-91.

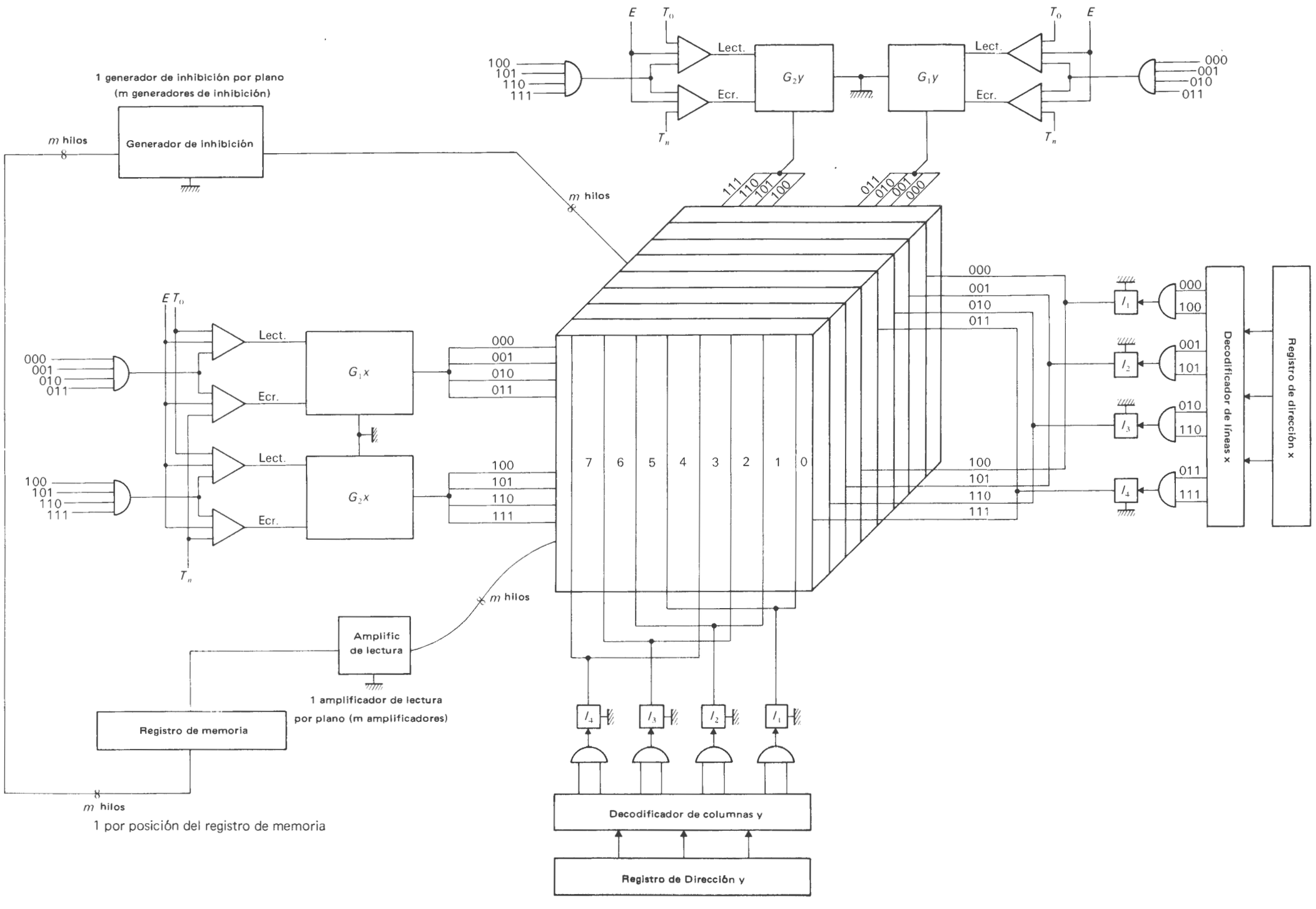


Fig. 11. *Sistema de direccionamiento y selección de palabras en la memoria por codificación de los interruptores I y de los generadores G. La memoria contiene 64 palabras de m bits*